A SENSOR-BASED INTERACTIVE DIGITAL INSTALLATION SYSTEM FOR VIRTUAL PAINTING USING MAX/MSP/JITTER

A Thesis

by

ANNA GRACIELA ARENAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2008

Major Subject: Visualization Sciences

A SENSOR-BASED INTERACTIVE DIGITAL INSTALLATION SYSTEM FOR VIRTUAL PAINTING USING MAX/MSP/JITTER

A Thesis

by

ANNA GRACIELA ARENAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Karen Hillier
Committee Members,	Carol Lafayette
	Jeff Morris
	Yauger Williams
Head of Department,	Tim McLaughlin

December 2008

Major Subject: Visualization Sciences

ABSTRACT

A Sensor-Based Interactive Digital Installation System
for Virtual Painting Using MAX/MSP/Jitter. (December 2008)
Anna Graciela Arenas, B.S., Texas A&M University
Chair of Advisory Committee: Prof. Karen Hillier

Interactive art is rapidly becoming a part of cosmopolitan society through public displays, video games, and art exhibits. It is a means of exploring the connections between our physical bodies and the virtual world. However, a sense of disconnection often exists between the users and technology because users are driving actions within an environment from which they are physically separated. This research involves the creation of a custom interactive, immersive, and real-time video-based mark-making installation as public art. Using a variety of input devices including video cameras, sensors, and special lighting, a painterly mark-making experience is contemporized, enabling the participant to immerse himself in a world he helps create. This work illustrates the potential of making the user-technology disconnection more seamless between the physical and virtual worlds. Using unobtrusive interfaces, the user's physical interactions can be encouraged. The development of this installation progressed through improvements based on user feedback from iterative public displays of the work. This process is to serve as a guideline for other artists working in interactive media who are also exploring perceived intimacy in user interactions.

DEDICATION

To God and my loving family

ACKNOWLEDGEMENTS

I would like to recognize my committee chair, Professor Karen Hillier, for her patience and guidance as my thesis came into fruition. This project saw a variety of iterations, and her ability to identify my strengths and interests helped me to channel them into the creation of a worthwhile project. I have valued her artistic advice and encouragement every step of the way. Much gratitude is owed to my committee member Dr. Jeff Morris for introducing me to Max/MSP/Jitter, and for his countless hours of programming guidance. Many preliminary ideas for this thesis project were envisioned when I began collaborating with him on another interdisciplinary project, merging improvisational music with improvisational visualization. Thank you, Dr. Morris, for believing in me and giving me the opportunity to grow. I would also like to acknowledge Professor Carol Lafayette, my committee member, who offered constant support, including invaluable and insightful programming and aesthetic considerations during the programming process. The advice and guidance of Professor Yauger Williams, my committee member, was an essential part of establishing the aesthetics of the final product. The passion and dedication of my entire committee has been truly unique and unparalleled; thank you all.

The Visualization Laboratory has been a special place for me over the last few years. I especially appreciate the friendship, support, and help of Tim Weaver, Karessa Bowens, Igor Kraguljac, Brooke Beane, Glen Vigus, and Kevin Glueck throughout the course of this research.

My time at Texas A&M University has certainly been a memorable one, and will be dearly missed. For all of the people I have met throughout the years, for your guidance and support in all my walks of life, thank you. I would especially like to thank those in Freshman Leaders in Progress (FLiP) for their enduring support and constantly reminding me year after year that I am still always a freshman at heart. For those I met through Aggie Awakening, Fish Camp and the Residence Hall Association, thank you for the memories. I would like to especially thank those at St. Mary's Catholic Church—including Fr. Brian, Fr. David, Fr. Keith, and Fr. Mike—for providing a special place of spiritual growth and renewal over my time here. *Ad Jesum per Mariam*!

No thanks would be complete without the acknowledgement of the friends I hold dearest to my heart for their total support and love: Jacob Foshee, Christian Clem, Charlotte McNabb, Austin Main, and especially Willem Strittmatter.

Last, but certainly not least, I would like to acknowledge my extraordinary family, especially my parents, to whom I am forever indebted for supporting me in all of my academic endeavors. To my father, who has been my light and inspiration through his life's example. To my mother, who taught me that all things are possible with God. To my older brother, Espiri, for the countless hours of conversation we have had, and unceasing support for my endeavors—especially related to this research. To my younger brother, Gabriel, for all of his enthusiasm for my work and for all of the laughs we have had along life's journey. My family's love and encouragement has truly been a blessing from God.

NOMENCLATURE

3D	Three-dimensional
API	Application Programming Interface
CPU	Central Processing Unit
DOF	Degrees of Freedom
GPU	Graphics Processing Unit
IEEE	Institute of Electrical and Electronics Engineers
IR	Infrared
LED	Light-Emitting Diode
MIDI	Musical Instrument Digital Interface
MSP	Max Signal Processing
OpenGL	Open Graphics Library
OSC	Open Sound Control
PC	Personal Computer
PVC	Polyvinyl Chloride
RAM	Random Access Memory
URL	Uniform Resource Locator
USB	Universal Serial Bus
USD	United States Dollar

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Currently, interactive digital media is woven into the fabric of every cosmopolitan society. Digital media has become such an integral component of everyday life, it may be difficult for some to recall what our society was like before its influx. There are frequent innovations in communications technology, including cellular phones, portable music and video devices, cameras, and video game consoles. Real-time interaction has become a key technological consideration in fast-paced societies that rely on instant aural, communicative, and visual satisfaction. It is this interactivity that acts as a bridge for humans to the digital world, enabling us to become more immersed in, and thus integrated with, the technologies we use.

Artist Camille Utterback writes, “the interactive medium provides a rich element to explore the connections between physical bodies and the myriad of representational systems possible in the digital realm” [1]. Indeed, the interactive medium encourages us as participants to engage as a bridge between the physical and virtual worlds. It invites us to employ human movement and the human senses of touch, sight, and/or hearing—sensations we rely upon heavily in real world actions—to facilitate immersion within the medium.

Interactive art installations are being introduced into greater society through means such as mall displays, video games, museums, and other art exhibits. Large

---

This thesis follows the style and format of *IEEE Transactions on Visualization and Computer Graphics.*

technology companies are exploring the line between the virtual world and the physical world with innovations like Microsoft Surface (Figure 1), which enables users to grab data with his/her hands and move information between virtual objects with natural gestures using a 30-inch table display [2]. The interface "disappears" in these virtual environments, as little to no equipment is required for a user to interact with them.

Figure 1. Microsoft Surface user interface, 2008 [3].

The more unnoticed (or unobtrusive) a user interface is, the easier it is for the average user to approach and familiarize himself/herself with it. Nonetheless, the dichotomy of the technology and the user can still be apparent. The user's actions drive the installation, for example, but s/he might still feel disjointed from the virtual world created by it. This feeling can be incited when playing video games: a gamer controls actions within an environment from which s/he is physically separated. A cause of this could be due to lack of intimacy and uniqueness of the interactions, resulting in the gamer's sense of indifference with the virtual world because s/he is not a part of this alternate reality. Games with first-person perspectives, such as Bungie Studios' *Halo 3*, are popular because they simulate affectable alternate realities (Figure 2). However, the inability of the gamer to employ his/her own human senses or have a realistic physical presence within this world could hinder the feeling of immersion in and intimacy with the medium.

Figure 2. Screenshot from *Halo 3*, Bungie Studios, 2007 [4].

1.1 Artistic Intent

To explore the implications of personalized interactive media, I propose to create an interactive, immersive, and customizable mark-making based real-time video installation as public art. Mark-making of a user in the installation space is contemporized using digital simulations, data inputs, and an unobtrusive user interface to create a personalized statement with painterly marks. The digital painting responds to the physical locations of the user within the installation and results in visual imagery of the participant "immersed" among his/her marks, suggested through a large-scale, user-viewable projection of the virtual world within the installation space. By investigating interactive possibilities with digital media, bridges between the physical and virtual world and the implications of personalized media will be explored.

1.2 Goals

The primary goal of this thesis is to create a public mark-making art installation with an unobtrusive interface. I would like to promote the idea of isolating and breaking down the dichotomy of user and technology through the use of physical actions—such as touch, body movement, and gesture— for personal interactivity. In order to make the installation more seamless with the physical world, I intend to create an efficient governing system based on real-time processing. My process is to serve as documentation for other artists interested in interactivity and working in a similar manner.

1.3 Artistic Impetus

Mark-making is an art form that has been used as a manner of expression since prehistory. In its various forms—from cave paintings to murals and even graffiti—it can be identified as art “of the people”. The marks used to make such forms of expression assert a presence or identity typically laced with personal statements. The democratic roots of this art are reinforced in its public visibility on legal (and illegal) public spaces and surfaces. These statements are crucial because, through their visual and/or typographical content, they speak to others, inciting thoughts and prompting an active (or passive) response. Although other aspects such as location and scale of the art can enhance these expressions, it is its message that is most crucial to reaching influential constituencies.

Figure 3. Cave paintings, the historical foundation of contemporary murals [5].

Spanish artist Joan Miró once said that he desired “through painting to get closer to the masses of humanity” [6]. Murals are democratic public art: they are placed in public places, available to anyone, and have the ability to speak to anyone who actively chooses to see and interpret their messages. They are, in a broad sense, “any form of large-scale articulate wall painting, mounted in public places, indoors or out, for viewing by large numbers of people at one time” [7]. In an historical context, murals have been used as public visual provocations for political and social commentary, conveying ideas to society without regard to literacy. The expression of this art relies on the articulate mark-making of a creative individual. It is a calculated form of expression that is typically narrative-based. Prehistoric cave paintings, for example, may have been created for reasons related to beliefs, good hunting fortune, rule delineation, or used for memory aids such as historical documentation or hunting methods (Figure 3). Traditionally, institutions, especially those religiously, socially, or politically based, and the artists supporting them, promote ideals that weave historical context and themes of empowerment into imagery. In the early Byzantine era, the teachings of Christianity were depicted in large-scale pieces in churches and cathedrals because these works were able to inform a predominantly illiterate public of the principles of the faith [6]. Artists of the Mexican Muralist movement (principally 1920-1940)—including David Alfaro Siqueiros, José Clemente Orozco, and Diego Rivera—created politically charged works depicting progress and oppression of the indigenous peoples of Mexico [8]. The spirit of their work radiated about a literary core known as the Syndicate Manifesto (1923), which stated, in part:

> The art of the Mexican people is the greatest and most healthy spiritual expression in the world [and its] tradition our greatest possession. It is great because, being of the people, it is collective, and that is why our fundamental aesthetic goal is to socialize artistic expression, and tend to obliterate totally, individualism, which is bourgeois [9].

Murals that are created in this spirit of collectiveness truly put their power—their ability to provoke, question, teach, and/or invigorate—into the hands of the people it was made for: the people that view them.

Graffiti can also stir political and social commentary, but, unlike murals, thrives on exploiting the world as its canvas. It is less calculated, more spontaneous, illicit (vandalism), and primarily word based. It is visual communication that is usually limited to the initials or name of the writer, a declaration of love or hate, or an insult, epithet, or political demand [7]. Graffiti, in its truest form, implies “alienation, discontentment, marginality, repression, resentment, and rebellion” [10]. Graffiti is a creation of those who have little representation within conventional mass media. “Hip-hop” graffiti is of particular interest to this artist. It is a realm of graffiti known for embracing a spirit of creation unmotivated by deliberate vandalism (although it is a byproduct of the work). Two forms of hip-hop graffiti that are of interest are “tags” and “pieces”: the former referring to single-line writings of an artist’s nickname and the latter referring to complex, multicolored pieces with intricate and decorative designs (Figure 4). These forms of graffiti focus on aesthetics and the overall quality of the marks. Like murals, they are assertions of the identity of people, they present an opportunity to share cultural values, and they are capable of redefining spaces. Author Alan W. Barnett eloquently writes that graffiti and murals “are types of struggle art by which people seek to survive as human in an increasingly dehumanized world" [7]. In this quest for survival, the marks of graffiti have the ability to provoke and question, while establishing an identity that desires to be recognized by humanity on some level.

Figure 4. Hip hop graffiti.

1.3.1 Personal Motivation

My interest in interactive, immersive installation art stems from a fascination with murals and mural creation as a teenager. Since that time, I was involved in the creation of three personal murals in public spaces. The ability to create a work of art with social implications helped me to understand the power of this art form. The individual images and articulate markings that make up a mural are as significant as the unified piece. The idea of personal mark-making gave way to a personal interest in graffiti and tagging as a form of individual expression. I became interested in the sense of individuality and liberation associated with personal mark-making. In this context, an artistic tool such as a paintbrush or marker symbolizes freedom to believe, think, and express. That is where the commentary begins. These thoughts and interests were brought together when I began to consider the concept of a virtual mark-making experience based on my background in art and computer science.

2. STATE OF THE ART

Innovations in real-time interactive and/or immersive media are found within various disciplines, including music, dance, theater, film, and visual arts. Many of these innovations are *intermedia*, integrating these disciplines. Previous work that is of relevance to this thesis research can be placed in one of five main categories: gesture-based performance pieces, movement-based installation pieces, augmented reality, video games, and interactive displays. These innovations, much like murals and graffiti, have the capability to redefine spaces (both physical and virtual) and put the power of interpretation and/or action into the hands of the people they are created for.

2.1 Gesture-Based Performance Pieces

2.1.1 Troika Ranch

Movement-based image generation has also provided an avenue for new forms of interactive expression. Troika Ranch, a digital dance company in New York City, has extensively explored the intermedia potential of dance, stage, and art technology. Founded in 1990 by choreographer Dawn Stoppicello and composer/media artist Mark Coniglio, Troika Ranch focuses on "creation, education, and innovation in theatrical performance" [11]. Specifically, the company's original works focus on the interaction between performers and digital media.

In 2006, Troika Ranch's production *[16]Revolutions* used live camera tracking technology to translate the movement of dancers on stage to manipulate digital media as real-time 3D imagery [12]. In this production, Coniglio wanted to "put power back in the hands of the performers" by giving life to what he called "dead media"—that is, media that is unchanging or fixed in nature [12]. Troika Ranch, a modestly sized group, typically uses dead media—particularly pre-recorded music on compact discs—in their performances. In order to make this media "give", Coniglio makes a different type of interactivity possible by using software that can respond to the environment, such as MAX/MSP/Jitter (see section 3.1.2.1) or Isadora. In [16]Revolutions, Coniglio employs a technique from artist Zachary Lieberman (see section 2.3.2) for motion tracking [11]. The stage environment consists of a large projection screen, or *cyclorama*, flooded with infrared (IR) light and a single IR camera positioned downstage. As the dancers move in front of the cyclorama, the camera sees a black silhouette (Figure 5).

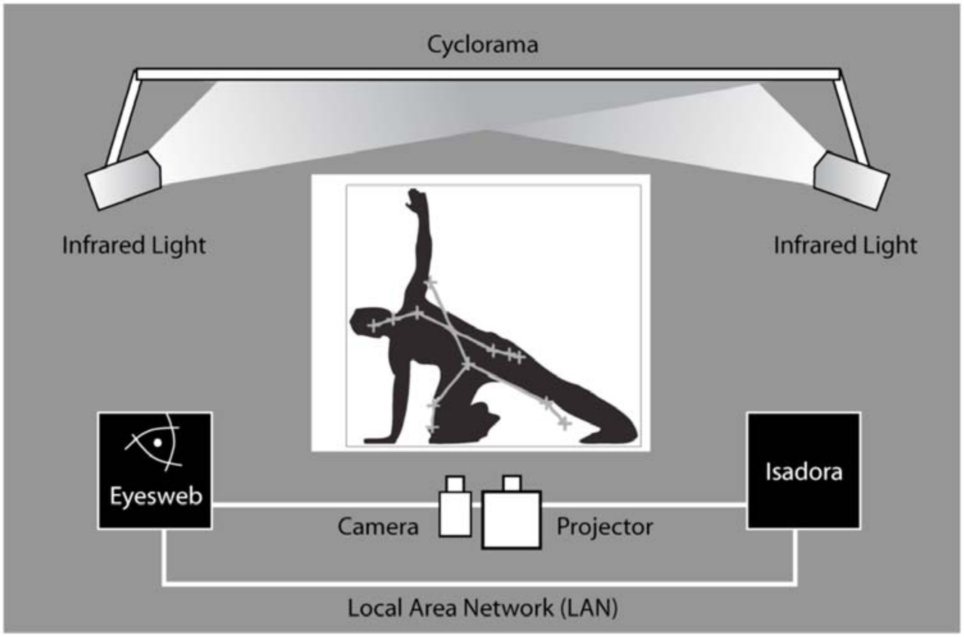


Figure 5. The silhouette-generation process of [16]Revolutions.

The silhouette is analyzed using a program called Eyesweb, which generates the coordinates of a 12-point skeleton (corresponding to torso and limbs) and sends these positions and trajectories to another program (Isadora®) for interpretation. Aspects of this data such as straightness, curvature, complexity, path length, and velocity are used to generate real-time imagery and sound. This visual product is projected on-stage with the dancers, creating an intimate relationship between the performer and the media. In the production, the physical movements of the dancers are enhanced, suggesting elements such as DNA strands or the expansion and contraction of the rib cage while breathing. This final product unifies the stage with the performer, establishing a visual and aural harmony with live human motion (Figure 6).

Figure 6. Troika Ranch in [16]Revolutions, 2006 [11].

2.1.2 Envyloop

The combination of sound and visual art in the context of live performances has provided new avenues for interactive multimedia expression. Envyloop, a “dynamic live-electro-improv-acoustic-meltdown duo” formed by musicians Ulrich Maiss and Butch Rovan, incorporates video into their live improvisational performances. These musicians create a unique and visually charged musical performance by employing a combination of acoustic instruments, found objects, interactive video, eclectic musical styles, and computer-aided processing elements (Figure 7).

Figure 7. Still from an Envyloop performance [13].

In a performance of Envyloop's program "Bleak Texas Thing" at Texas A&M University in October 2007, approximately 80 random prerecorded video clips were placed into a database and then coordinated using a program called *Grid* for effects processing. This program was used to coordinate the timing of, queue, and transition between these clips in real-time. This visual product was projected on a screen behind the musicians during the performance. A few minor events, such as instrument movement, slightly affected the video output in real-time, but the video processor was not designed to be wholly congruent with the spontaneous movement of the instruments or musicians on stage. The pairing of Envyloop's real-time audio processing with these visuals resulted in rich, interwoven sensory textures. Like Troika Ranch, this group's work is significant evidence of the synergistic potential of intermedia efforts. This results in a type of elevated media: a creative product that is stronger than the singular elements that comprise it.

2.1.3 Research with Jeff Morris (Video Improvisation Project)

Real-time and mediatized (processed) aural and visual events in improvisational musical performances were explored in this artist's work with Dr. Jeff Morris, Eric km Clark (electric violin), and Andy McWain (pianist). In this iterative collaboration, called *Time is the Substance of Which I Am Made*, an improvisational violin and piano performance was enhanced with video and sound-processing elements [14]. Dr. Morris facilitated the mediatization of sound using an original gesture-based program driven by a Nintendo wiimote. The accompanying visuals were produced in a similar manner.

This artist drove the real-time response of the visuals to the music and guided video transition events based on the observed tempo of the piece and movements of the musicians. The sensor data from the wiimote affected video filter parameters, combination techniques, and playback controls in real-time. This empowered this artist to become a “visual musician” by making familiar artistic gestures to create a visual product.

The visual work in this collaboration extends the music visualization ideals presented in Envyloop’s “Bleak Texas Thing” to invite the audience to participate in a real-time visual improvisation that parallels its aural counterpart. The video is shot specifically by this artist on-stage to capture and emphasize gesture events (especially those normally overlooked during a performance) such as bobs of a musician’s head in response to playing an instrument or the movement of a violin bow as it glides across the instrument’s bridge. This information is processed and then projected onto the performers and a screen on the performance stage, allowing the filtered video to be captured by the video camera and reprocessed. This enhances the visual product by adding unique and improvisational qualities and giving the complete performance work an aural and visual cohesiveness. Video stills of the group’s performance at Northwestern University (International Society for Improvised Music Conference, December 2007) are illustrated in Figure 8: the left side is the original footage, while the right side is the real-time filtered video that resulted from the original.

A real-time sketching feature was added to the video-processing program in a later iteration of the work. This feature enabled this artist to draw on top of the live, processed video using the wiimote, adding another dimension to the improvisational quality of the visuals. The development of this feature laid important groundwork for the foundations of this thesis research (see section 4.2.2.2).

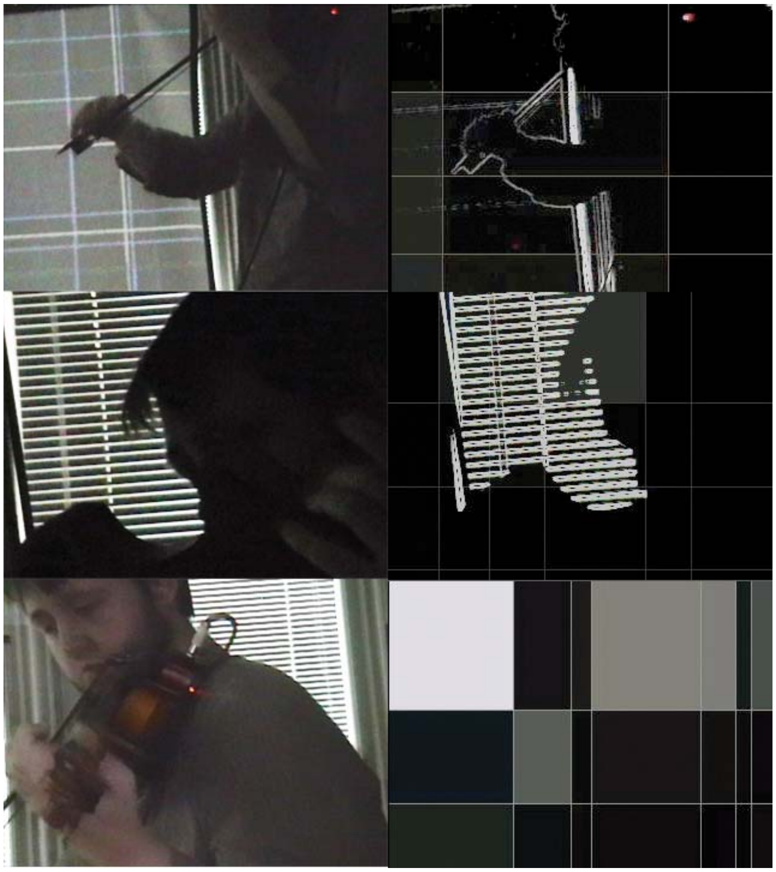


Figure 8. Before-and-after video stills, International Society
for Improvised Music Conference, 2007 [14].

2.2 Movement-Based Installation Pieces

2.2.1 Camille Utterback

Interactive video artist Camille Utterback, has created a number of works related to movement-based image generation. Her installation pieces respond to a participant's location in the space, spatial relationships among multiple participants, and gesture. In her essay "Unusual Positions — Embodied Interaction with Symbolic Spaces", Utterback writes:

> With much of my artwork — in both traditional and digital media — I have attempted to draw attention to the connections between human bodies and the symbolic systems our bodies engage with. The digital medium interests me because it is a perfect site to explore the interface between physical bodies and various representational systems…How and to what extent new interfaces may engage the body…is up for grabs. [15]

This mindset of engaging the body in the digital medium has fostered the creation of many unique works. Utterback's *Eternal Measures* series, for example, focuses on aesthetic systems that "respond fluidly and intriguingly to physical movement in the exhibition space" [1]. *Untitled5* (2004), a work in this series, creates organic, painterly, and algorithmic marks as it maps the trajectory of a participant in the installation space (Figure 9) [16]. Custom software is used to generate this image from a "palette" of animated marks connected to the viewer's movements in various ways. Over time, these marks are cumulative and can be pushed from their original locations by other people's movements in the space. An interesting connection between time and space is created when these displaced marks try to move back to their original locations, resulting in streaks of color. The resulting artwork has the mark of many participants over time. Utterback's systems are simple, involving no complicated apparatuses: it is simply the congruence of the participant and the media in the presence of each other. This adds an appealing element of approachability to her work, as participants can learn how to use the spaces she creates by simply being present in them.

Figure 9. *Untitled 5*, Camille Utterback, 2004 [1].

A common characteristic among Utterback's work is that the participant undergoes a transformation that places him/her in an alternate reality, inviting him/her to explore the feeling of being a part of it. In *Text Rain* (1999), for example, participants view a projection of themselves that is capable of catching, lifting, or dropping falling text (Figure 10). Participants stand in front of a projection screen on which they see a mirrored "alter ego" that can interact with a series of falling letters that spell out the lines of a poem. In the context of this installation, text behaves like real-world objects, responding to forces in the physical world. Utterback writes:

> Because most of one's body is visible in the virtual space of the screen as well as in the physical space in front of the screen, a pleasurable confusion results between the screen space and the real space. Because no complicated apparatus is involved to become "immersed" you can easily feel present in both the physical and virtual space simultaneously, or seamlessly shift back and forth between the two [15].

This sense of immersion is unique because it has the capability to bridge the physical and virtual worlds to create a sense of being in a hyperrealistic realm.

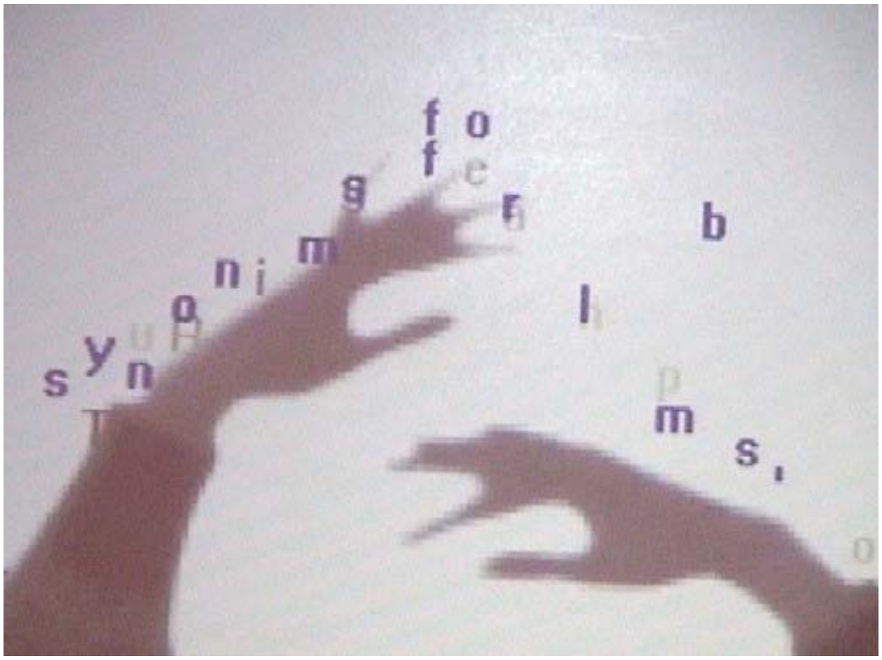


Figure 10. *Text Rain*, Camille Utterback and Romy Achituv, 1999 [1].

2.2.2 Daniel Rozin

Like Utterback, artist Daniel Rozin has also investigated the field of interactive digital art with pieces that have a one-of-a-kind ability to "change and respond to the presence and point of view of a participant" [17]. His work titled *Wooden Mirror* (1999) exploits the organic qualities of pieces wood to act as digital pixels based on their orientation in relation to the viewer (Figure 11) [18]. A small camera in the center of the mirror sends images to a computer that are processed to send signals to motors that orient the pieces of wood. These wooden tiles reflect different amounts of light based on their degree of tilt (towards or away) from a main light source. In this way, participants viewing the unit of wooden surfaces can see their rough digital image "reflected" back to them. As in Utterback's work, Rozin creates a simple interface that honors the congruence of the participant and the media in the presence of each other, only requiring a participant to be within the camera's field of view. Although the abilities of Rozin's and Utterback's interactive art are executed using digital technologies, these aspects are hidden to encourage the participant to focus on the aesthetic quality and creation of the work.

Figure 11. *Wooden Mirror*, Daniel Rozin, 1999 [17].

2.3 Augmented Reality

Innovations in the field of augmented reality (AR) add virtual information to a user's sensory perceptions, including sight, sound, and/or touch, using computer displays. AR systems typically involve the use of optical see-through displays that overlay graphics on the wearer's view of his/her surroundings as the position and orientation of his/her head is tracked. Rather than aiming to replace our physical world with virtual representations, as in virtual reality, AR aims to complement it [19].

2.3.1 Jung von Matt /next

Jung von Matt /next, an agency for interactive and innovative communication in Hamburg, Germany, is the developer of the *Tagged in Motion* project, a virtual graffiti-creating experience. In this work, three video cameras surround the artist and are used to record the position of a pattern affixed to the artist's forehead (to determine his/her line of sight) and a virtual spray can that s/he controls (Figure 12). These positions are calculated using a pattern recognition feature of ARToolKit (Augmented Reality ToolKit), an open-source software library for building augmented reality applications. The position of the virtual spray can is translated to 3D space to create "floating" graffiti in real-time. The quality of this line is modified using a Bluetooth controller that can affect parameters such as texturing, colors, and line thickness [20]. This data is viewable to the artist with the help of special video glasses that use his/her head position to render the generated graffiti in the correct perspective. Although this work requires a moderate amount of user interface equipment, the developers of this project have created an innovative, immersive, and individualized artistic experience.

Figure 12. Still (with simulated marks) of the *Tagged in Motion* project, 2008 [20].

2.3.2 Zachary Lieberman

Artist and programmer Zachary Lieberman creates innovative "playful" technologies that are designed to investigate communication through gesture, body augmentation, and kinetic response. He is the co-creator of an open source C++ library called openFrameworks, which is intended for technical-minded artists who desire to use computers in creative forms of artistic expression. Lieberman uses this library in his performance/installation work titled *drawn* (2005) to create a live painting that is "radically augmented in real time, creating a fictional world in which the painted forms appear to come to life, rising themselves off the page and interacting with the outside world" [21]. The interface is simple, consisting of a table with paper, ink, and brushes. An overhead camera captures the image as it is drawn, sending this information to a computer that transforms the marks into synthetic graphics using a complex algorithm. A hybrid video signal is created to combine these artificial marks with the drawing environment. The artist can then interact with their marks and create music by moving them within the page with various hand movements (Figure 13). In this real-time fictional world, the marks are transformed into kinetic forms. This "visual instrument" defines a poetic relationship between the physical and virtual worlds.

Figure 13. *drawn*, Zachary Lieberman, 2005 [21].

2.4 Personal Interactivity in Video Games

2.4.1 Electronic Arts

Avatars in video games, as described in the introduction, usually offer a degree of customization from a predetermined set of character types and features, providing little to no opportunity to fully personalize a character with a player's likeness. An extension of this idea, developed by gaming company Electronic Arts (EA), has added a deeper element of personalization in video games. This technology, known as Game

Face, was first used in Tiger Woods PGA Tour 2004 [22] and enabled gamers to create personalized avatars of themselves using a character creation application, complete with customizable aspects such as hair, skin type, body type, and facial features. This unique character can then be played directly in the game. An additional feature, called Photo Game Face, was added to this existing structure in Tiger Woods PGA Tour 2008 for the Xbox 360 and Playstation 3 consoles [23]. This application makes it possible for a gamer to simply upload up to two pictures—a front and a profile view of his/her face—to create an even more true-to-life character (Figure 14). The Game Face features facilitate an innovative, more intimate kind of gameplay experience that directly invites gamers to immerse themselves in (and become a part of) the virtual world.

Figure 14. Photo Game Face, Electronic Arts, 2007 [23].

2.5 Interactive Displays

2.5.1 GestureTek

The interactive, real-time advertising and gaming interfaces of GestureTek Inc™ respond to movements and gestures of participants in public spaces such as malls, movie theaters, and museums. Using proprietary body-tracking software combined with IR lighting and video camera technology, participants can control a user interface projected onto a floor or wall or displayed on a plasma screen. In some implementations, such as in GestureTek's GestureXtreme™ and ScreenXtreme™ series, users are captured and inserted into virtual interactive environments as real-time video. These displays enable users to participate in events such as a virtual sporting games (Figure 15) and virtual object displacement in advertisements (Figure 16). Although the user can affect the environment they are integrated into, s/he cannot personally determine the visual makeup of the environment itself. The dynamic interactivity of these displays is implemented without the need for participants to wear, touch, or hold anything [24], making it a perfect candidate for any public space.

Figure 15. GestureXtreme™ interactive gaming system, GestureTek, 2008 [25].

Figure 16. ScreenXtreme™ interactive advertising system, GestureTek, 2008 [25].

2.5.2 Graffiti Research Lab

The endeavors of the Graffiti Research Lab in New York propel the implications of graffiti to modern-day proportions by integrating elements of digital technology into its creation. Founded by Evan Roth and James Powderly, the lab is primarily interested in developing open source graffiti technologies that transform everyday spaces into artistic ones [26] and outfitting graffiti artists with these innovations for use in urban communication. Each graffiti endeavor is recorded on video and step-by-step documentation (including code) is freely available on their website.

Figure 17. *Laser Tag* project, The Graffiti Research Lab, 2007 [27].

Their project *Laser Tag* (2007), involved a camera and laptop setup that tracked a high-power green laser point across the face of a building. This positional information was used to generate graphics that were then projected onto the same building using a high power projector (Figure 17). A C++ application using openFrameworks (see section 2.3.2) was used to align the camera to the projection surface and adjust settings related to laser point detection and line quality (including a “dripping” effect that gives the marks a sense of fresh paint) [27]. The end effect was a laser pointer that enabled artists to create large-scale graffiti-like marks on the sides of a large edifice. This type of artwork fosters an exceptional interaction between artists and large-scale public spaces that is unachievable with traditional media. In addition, this work suggests immersion through the sheer scale of the projected work on a physical (real world) environmental surface.

2.5.3 Johnny Chung Lee

Johnny Chung Lee received widespread acclaim on video-sharing website YouTube in 2007 with his instructional videos on inexpensive wiimote-based (see section 3.1.2.2) interactive works. These open source innovations are devised from simple methods to handle complex tasks such as finger and head tracking. Through his cost-effective methods, Lee aims to make technology accessible to a much wider percentage of the population [28].

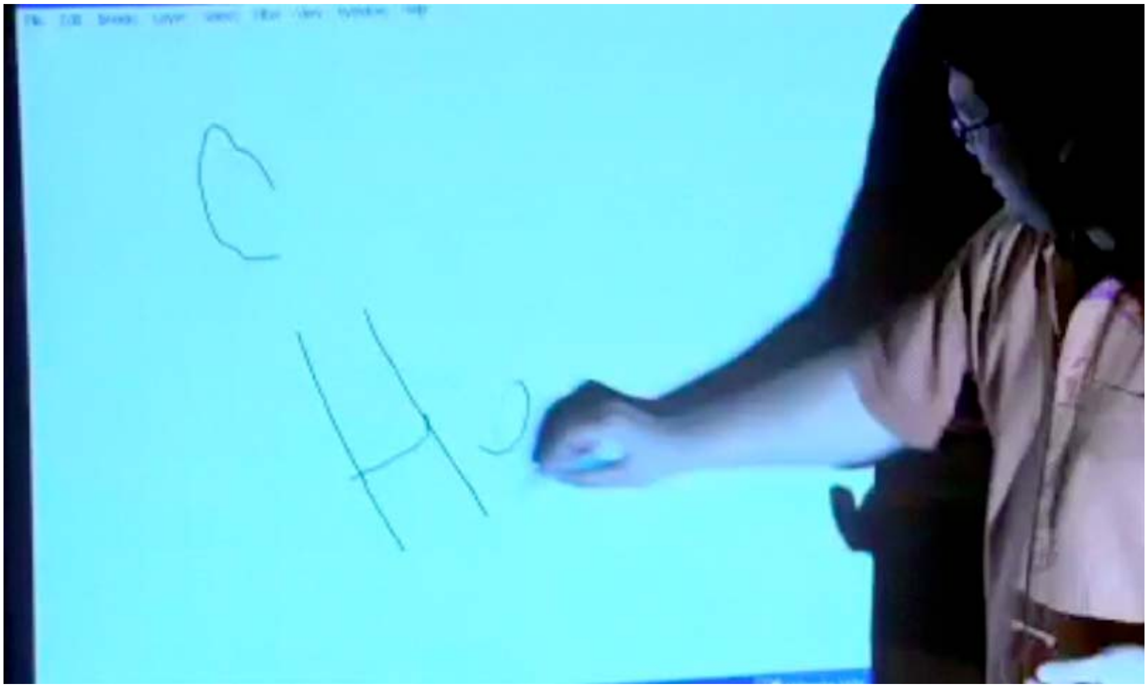


Figure 18. *Low-Cost Multi-point Interactive Whiteboards Using the Wiimote*, Johnny Chung Lee, 2007 [28].

In his work *Low-Cost Multi-point Interactive Whiteboards Using the Wiimote* (2007), Lee presents a technique for creating an interactive touch screen interface using a projector or an LCD touch screen. Using C# programming language-based software, the IR camera of a wiimote is used to track the movement of an IR-emitting pen in its field of view. After a calibration step to register the location of the camera to the projected (or LCD screen) pixels, a surface can be transformed into an interactive whiteboard. The IR pen can simulate a mouse, enabling it to perform as a writing and control tool in computer applications such as Adobe Photoshop (Figure 18). The wiimote has the capability to track up to four points, so up to four IR pens can be used on the whiteboard at one time. This whiteboard setup, at under $50, is a fraction of the cost of its industrial counterparts, promoting a more accessible method for this type of interactivity.

3. METHODOLOGY

An interactive, immersive, real-time video installation where the user makes gesture-based marks is the ultimate goal of this research. The final artwork is in the form of a real-time processed video of the user generated from a physical environment and incorporated amongst the user's marks. The main objective of this piece is to illustrate a heightened sense of immersion for the user, and therefore a feeling of increased seamlessness with digital media. For the artist, it is a point of exploration of the gap between user and media that suggests the bridging of these worlds by utilizing video cameras, IR lighting, and sensors.

The interactive and immersive works presented in the previous section include the following principal characteristics:

- participation of a user in a particular setting or environment,
- incorporation of technology (either blatant or hidden), and
- an application that determines how it will be made available to the public (such as in a mall, an art museum, or via a video game console).

Because my installation is also an interactive, immersive work, these overriding qualities were imperative in its construction. The approach to creating a robust, interactive, immersive, and real-time video installation required in-depth investigation. This included evaluating technical and aesthetic goals, speaking and working with other artists, constructing the physical installation space, and programming the visual interface. Working on each of these facets iteratively enabled me to effectively realize their combined workability for the final piece.

3.1 Phase 1: The Planning Process

3.1.1 Installation Constraints

There were a few constraints placed on the creation of the installation by this artist in order to guide its process. The installation space was to be easily accessible, simple to enter and exit, and large enough to promote arm and body gestures. The subject content of the work needed to be appropriate for all ages because of its intended display in public areas. The desired approach to immersion of the user in the installation was to directly place the user in a viewable and dynamically changing alternate reality. Although multiple people could be in the installation space at one time, it was to be developed optimally for an individual. The installation site was not to be site-specific, although it required an environment without intense IR-producing light sources. In addition, it was desired that the technical delivery systems of the work be obscured as much as possible to increase approachability of the work.

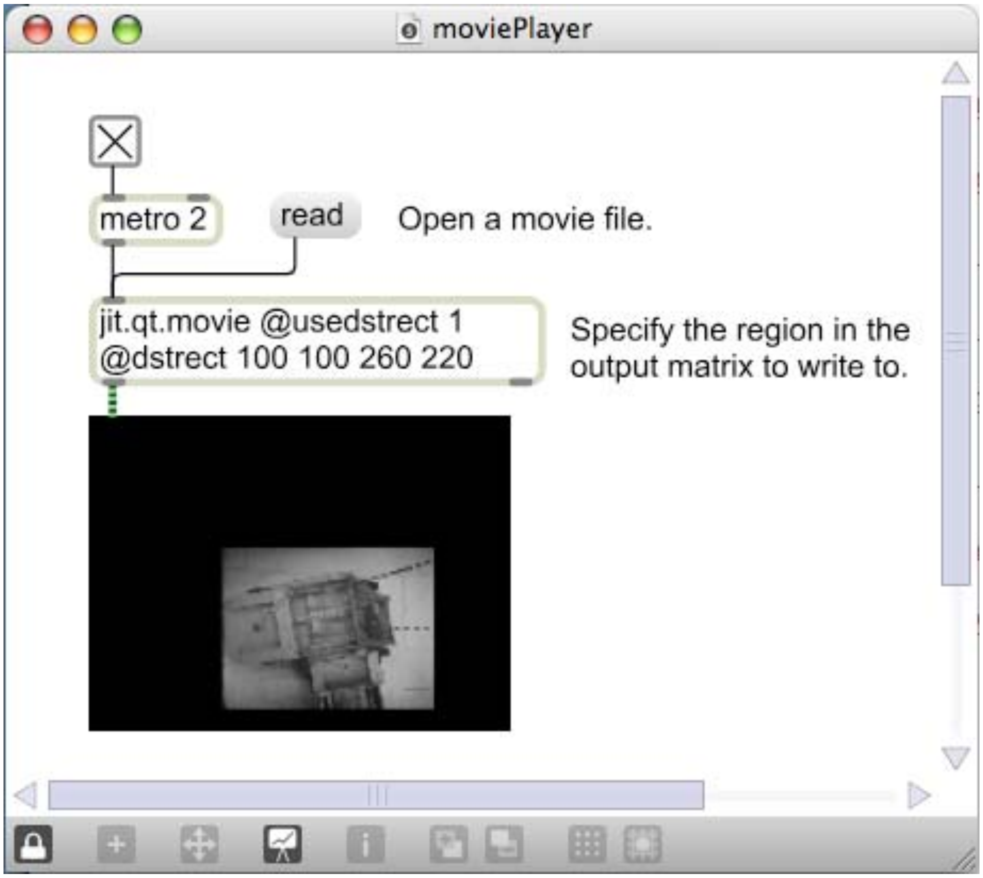
3.1.2 Technical Considerations

The “interactivity catalysts” of the installation—the programming and computing environment, sensor inputs, and environment lighting—required thoughtful (and sometimes iterative) consideration. The efficiency of these components, individually and collectively, was essential to an optimal construction of the installation and a high standard of visual quality.

3.1.2.1 Programming and Computing Environment

In this thesis project, the most important factor for creating real-time interactivity is the program that facilitates communication with the virtual world and produces instant visuals. The installation requires real-time responsive visuals based on gesture and spatial movement of the user. In order to meet these needs, the software used to create the program must be flexible, customizable, enable sensor interactivity, manage 3D graphics, and handle the incorporation of live video feeds. In addition, the programming environment needed to be extendable for the development of unique tasks. Most importantly, the software and hardware must be capable of the simultaneous real-time processing of data and video streams.

The software Max/MSP/Jitter is designed to handle such tasks, and consequently was chosen as the programming environment to facilitate the installation's interactivity components. Max/MSP is a current manifestation of a programming paradigm that has been used since the mid 1980s, first for MIDI automation and followed by real-time sound processing. In 2003, a set of objects known as "Jitter" expanded these capabilities to real-time video. In this powerful environment, programs, or *patches*, are created using a graphical interface (Figure 19). The visual programming style employed by Max/MSP/Jitter enables developers to easily create programs by connecting a series of graphical modules to create a flow of events that can be evaluated in real-time. This is in contrast to traditional programming methods, which require the writing of collective lines of code in order to perform tasks.



```
metro 2
read
jit.qt.movie @usedstrect 1
@dstrect 100 100 260 220
```

Open a movie file.

Specify the region in the output matrix to write to.

Figure 19. A MAX 5 patch (Max/MSP/Jitter) with video frame, 2008.

Although Max/MSP/Jitter has been ported for use on PC machines, Max was originally developed for Macintosh computers [29] and runs optimally in this environment. Therefore, this artist developed programming for the installation on a MacBook Pro using the Mac OS X 10.4.11 operating system. This particular computer was equipped with a 2.4 GHz Intel Core 2 Duo processor, an NVIDIA GeForce 8600M GT graphics processor, built-in Bluetooth 1.9 (for wireless communication), and 4 gigabytes of RAM. This artist also ensured that the most current versions of Quicktime and OpenGL were installed. All of these specifications provided adequate support for

Max/MSP/Jitter's performance requirements; however, this artist's process still required a careful balance of computing resources.

3.1.2.2 Interactivity Through Sensor Input

Another important factor for interactivity is sensor input. Artist Tom Igoe supports their use in physical computing (the creation of interactive physical systems), writing that "position and motion sensors make it relatively simple to take advantage of body position and movement" [30]. Sensors provide an avenue for a participant in an installation environment to communicate with digital media.

As previously mentioned, Max/MSP/Jitter is capable of interfacing with a variety of sensors. This includes devices such as accelerometers, pressure sensors, and distance sensors. These can be used to translate actions and movements into data that can be manipulated in Max/MSP/Jitter.

In order to establish the type of sensors needed for the installation, this artist determined what type of information needed to be gathered from the motions of participants in the space. The most essential factor of motion to consider was position within a limited zone. This artist needed to be able to calculate and track the following:

- an x-y cursor position to enable the participant to draw a line, and
- the z-position of a person in the space (in relation to the viewing screen) to facilitate immersion amongst his/her marks.

In addition, this artist needed to consider how to trigger specific events, such as initiating and suspending line drawing or erasing an existing line from the screen. A secondary factor of motion to consider was the orientation of the user-controlled “paintbrush” as the line was being drawn, which would affect the thickness of the line generated.

Initially, this artist wanted to incorporate many individual sensors in the installation, notably multiple distance and pressure sensors or ultrasonic sensors placed strategically in the space to calculate a person’s location. This early deliberation was soon reconsidered due to concerns of overloading the computer processor with concurrent streaming data from multiple sensors. A delicate balance was needed between software and hardware for efficient real-time processing. An optimal solution to this problem was to incorporate the Nintendo Wii Remote (or *wiimote*, Figure 20) into the installation. The wiimote is compact, handheld, wireless, comprised of multiple sensors and controls, can interact with Jitter using standard Bluetooth technology, and is well known from its traditional use in gaming. The wiimote is also extendible: additional sets of plug-in controls (such as the nunchuck or classic controller, see Figure c) allow for different types of interaction and expand the capabilities of the wiimote. Furthermore, the wiimote is an inexpensive device: at the time of this writing, it is available for purchase for about $40 USD.

Figure 20. The Nintendo Wii Remote, or *wiimote*.

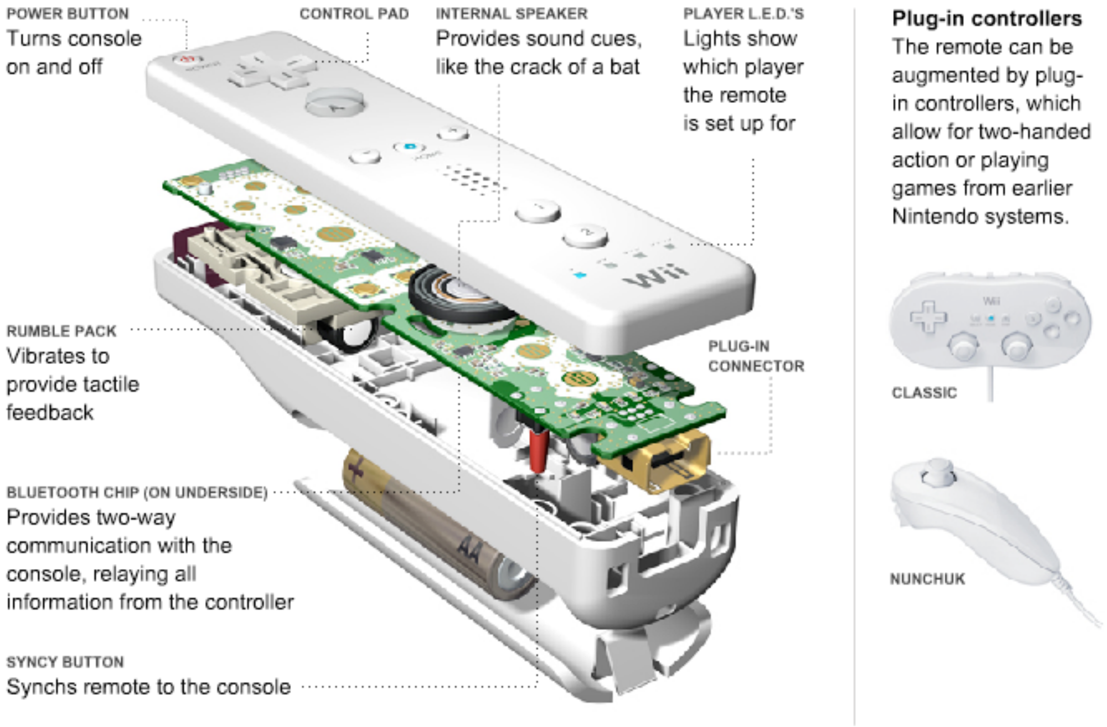


Figure 21. Built-in features of the wiimote and common plug-ins [31].

The wiimote consists of a triple-axis accelerometer, IR sensor, and twelve buttons—perfect for many of the installation's sensor-based needs (Figure 21). The accelerometer reports instantaneous acceleration imparted on the wiimote as it is being controlled. Six degrees of freedom (DOF) can be calculated from the accelerometer data (Figure 22): linear translations (x, y, z) and rotation angles about the x/y/z axes (pitch, roll, and yaw).

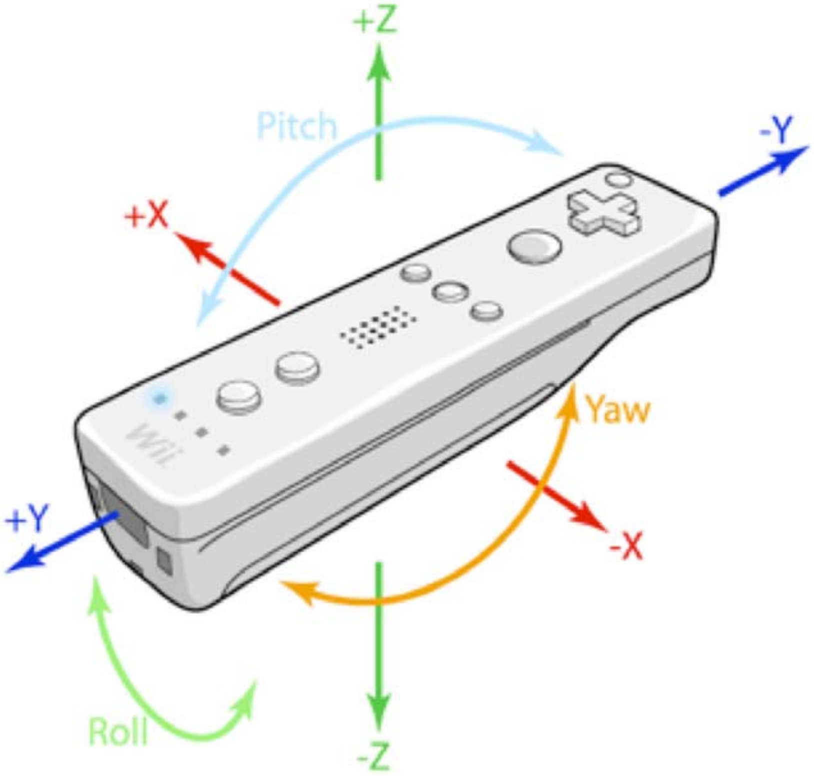


Figure 22. The six DOF calculated using the wiimote's accelerometer data [32].

The IR sensor, located at the tip of the wiimote, sees only IR light and enables the wiimote to also act as a pointing device. This feature is only functional when the wiimote's IR sensor is within range of IR emitters. A typical device used with the wiimote is a *sensor bar*: a shell that houses two sets of IR LED light sources—one on each end (Figure 23). This sensor bar is placed in a static location and oriented such that the IR sensor can detect both of the sensor bar's IR light sources within its narrow field of view (Figure 24 and Appendix A). The most effective working range of the wiimote in this setup is within a distance of about 10 feet from the sensor bar. The IR sensor can track the two IR light sources from the sensor bar in a two-dimensional plane to calculate approximate x-y pointing coordinates. It can also use further information from these two points to calculate a general distance from the light sources using triangulation (Appendix B).

A wired sensor bar comes with the Wii gaming console, but the console must power it. However, homemade ones can be constructed fairly easily using sets of IR LEDs arranged in the same manner. Sensor bars have even been fashioned by simply using common sources of IR light, such as candles, which can be spaced apart. This artist's use of the wiimote does not necessitate the use of the entire Wii console, so a portable, stand-alone battery-powered sensor bar manufactured by Nyko is used to obtain the same general functionality (Figure 25).

Figure 23. Wiimote pointing using the sensor bar [31].

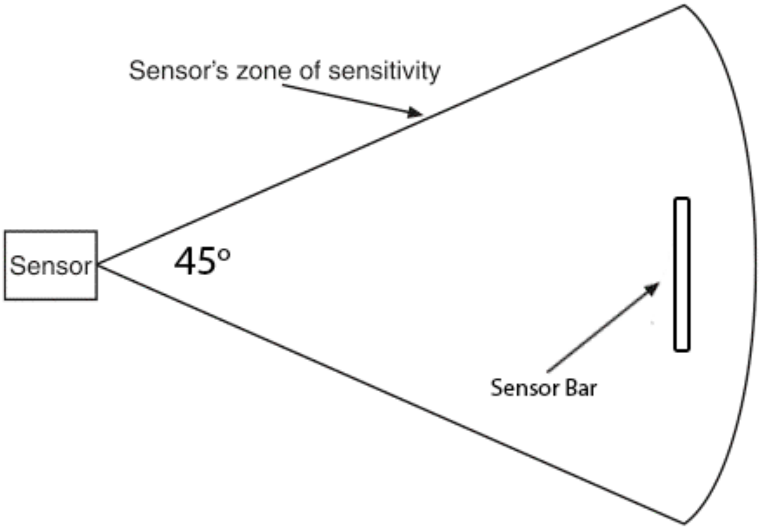


Figure 24. The range of visibility of the IR sensor on the wiimote.

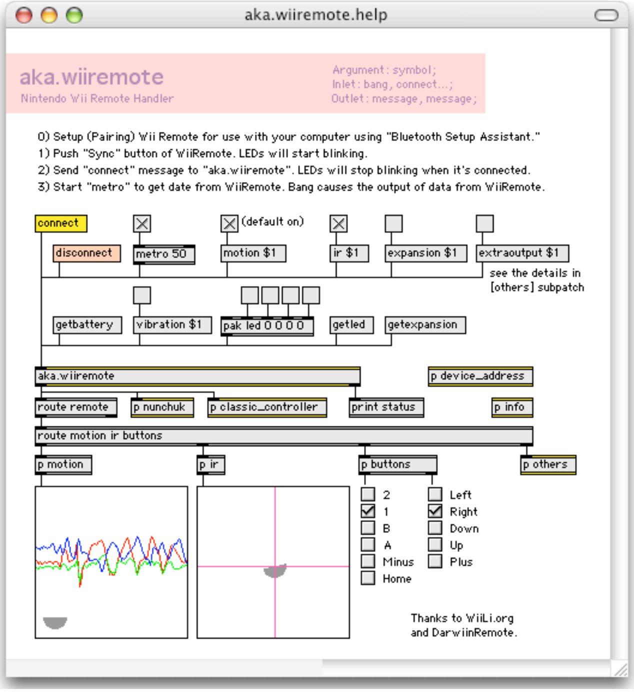
(a) (b)

Figure 25. Two types of sensor bars: the standard wired bar for use with the Nintendo Wii gaming console (a), and a stand-alone battery-operated bar (b).

3.1.2.3 Interfacing the Computing Environment with the Wiimote

In order to communicate with the computing environment, this artist explored a couple of options that would enable message transfer between the wiimote and Max/MSP/Jitter. Originally, this artist developed the wiimote functionality for use with an external Max object called aka.wiiremote (Figure 26), developed by Masayuki Akamatsu in 2007. It is an object based on DarwiinRemote, a Macintosh-based application that allows the wiimote to drive applications by employing most of its features. Although aka.wiiremote provided access to the wiimote's data, it was unstable: there were difficulties *pairing*, or connecting, the wiimote to the computer using Bluetooth or obtaining responses from the various sensors once the wiimote was connected. Getting the wiimote to work correctly required disconnecting the device and reconnecting it multiple times or simply restarting the computer.

Further research on wiimote connectivity to Max/MSP/Jitter revealed the existence of another application called OSCulator. Although this program can communicate with applications such as Max using MIDI (a protocol enabling musical instruments and multimedia devices to interface with each other) and keystrokes on the keyboard, it also uses an optimized networking-based Open Sound Control (OSC) protocol. This protocol enables musical instruments and other multimedia devices to communicate quickly over a network (such as Ethernet or the internet). OSC messages consist of hierarchical, URL-type addresses that can be sent at broadband network



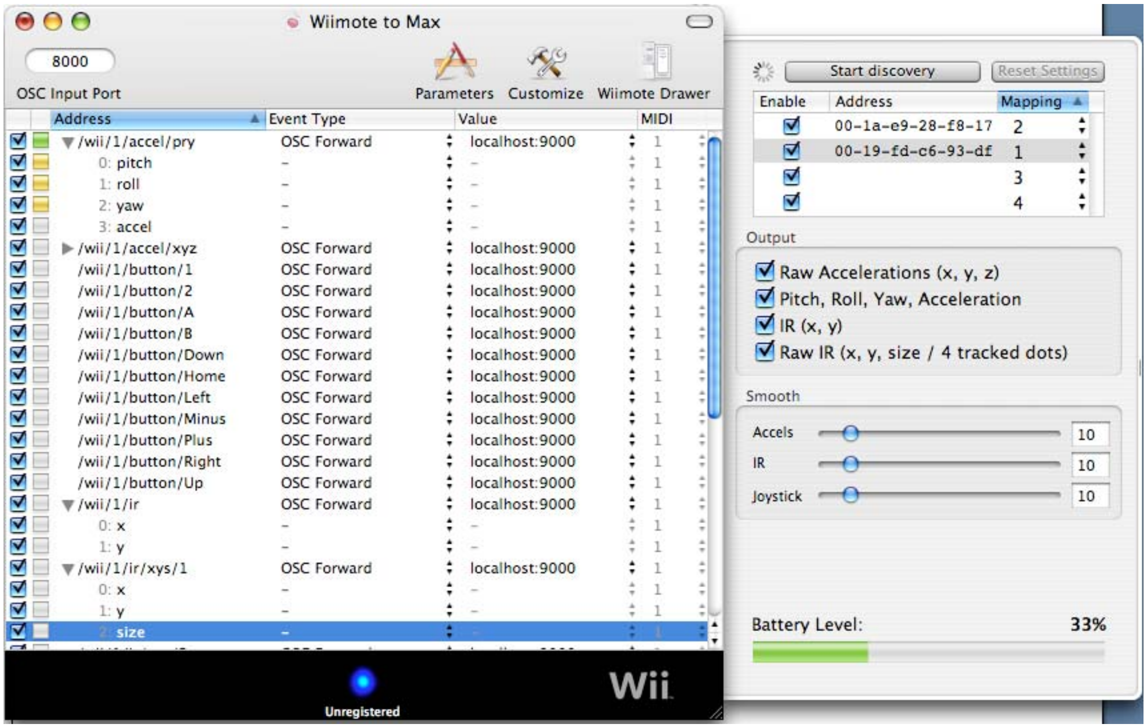
0) Setup (Pairing) Wii Remote for use with your computer using "Bluetooth Setup Assistant."
1) Push "Sync" button of WiiRemote. LEDs will start blinking.
2) Send "connect" message to "aka.wiiremote". LEDs will stop blinking when it's connected.
3) Start "metro" to get date from WiiRemote. Bang causes the output of data from WiiRemote.

Figure 26. The aka.wiiremote object for Max [33].

speeds—overall, much faster than MIDI. The addition of an OSC object to the Max/MSP/Jitter object set enables OSCulator to interface rather easily to this programming environment via a receiving port. The following information from the wiimote can be tracked [32]:

- Button events: binary value corresponding to the pressing or releasing of any button on the wiimote,
- Raw Accelerations (x, y, z): acceleration values as measured by the accelerometer chip of the wiimote,
- Pitch, Roll, Yaw, Acceleration: values that represent the orientation of the remote and, correspondingly, are derived from the x, y, and z accelerations and the scalar value (overall value) of {x, y, z},
- IR: represents the x and y coordinates of an imaginary point to which the wiimote is directed, and
- Raw IR (x, y, size / 4 tracked dots): values as given by the built-in IR camera. The wiimote can track up to four dots; their x-y coordinates are reported, as well as their sizes (calculated by triangulation).

All of this information can be channeled to Max by a process called OSC forwarding, while all accelerometer and IR data can be smoothed (optionally) for more steady readings. Furthermore, a "perfect pairing" technology feature, released in version 2.5.1, enables wiimotes to connect reliably to OSCulator, usually on the first try [32]. For these reasons, this artist found the OSCulator environment to be much more robust upon testing with Max/MSP/Jitter, and this artist consequently resolved to use it to handle the wiimote functionality in the installation.



OSC Input Port: 8000

		Address	Event Type	Value	MIDI
☑	🟩	▼ /wii/1/accel/pry	OSC Forward	localhost:9000	1
☑	🟨	0: pitch	–	–	1
☑	🟨	1: roll	–	–	1
☑	🟨	2: yaw	–	–	1
☑	☐	3: accel	–	–	1
☑	☐	▶ /wii/1/accel/xyz	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/1	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/2	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/A	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/B	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Down	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Home	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Left	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Minus	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Plus	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Right	OSC Forward	localhost:9000	1
☑	☐	/wii/1/button/Up	OSC Forward	localhost:9000	1
☑	☐	▼ /wii/1/ir	OSC Forward	localhost:9000	1
☑	☐	0: x	–	–	1
☑	☐	1: y	–	–	1
☑	☐	▼ /wii/1/ir/xys/1	OSC Forward	localhost:9000	1
☑	☐	0: x	–	–	1
☑	☐	1: y	–	–	1
☑	☐	2: size	–	–	1

Enable	Address	Mapping
☑	00-1a-e9-28-f8-17	2
☑	00-19-fd-c6-93-df	1
☑		3
☑		4

Output

☑ Raw Accelerations (x, y, z)
☑ Pitch, Roll, Yaw, Acceleration
☑ IR (x, y)
☑ Raw IR (x, y, size / 4 tracked dots)

Smooth

Accels: 10
IR: 10
Joystick: 10

Battery Level: 33%

Figure 27. A screenshot of OSCulator, after pairing with the wiimote.

3.1.2.4 Lighting in the Installation Environment

Lighting in the installation was an important consideration as it aids in extracting a participant from the environment, much like bluescreening techniques seen in television and movies today. Troika Ranch's approach for *[16]Revolutions* (section 2.1.1) was an elegant alternative to this method that did not cause artifacts typically found with bluescreening methods associated with blue spill and chromakeying. This approach was introduced to this artist in a discussion of this thesis research with Butch

Rovan of Envyloop (section 2.1.2). In the method, IR lights are flooded onto a cyclorama. IR light is invisible to the human eye, falling within a wavelength between 750nm and 1nm. However, it can be made visible to video cameras equipped with special lenses that can see only IR light. If such a camera is directed towards a participant standing in front of the IR-illuminated cyclorama, some IR light is prevented from being reflected back to the camera lens. The result is crisp, black and white, silhouetted real-time footage of the participant. Using this silhouette, a real-time video mask can be created (Figure 28). In this context, *masking* is a digital technique used to drop specific portions of video footage based on the opacity of the negative portions of black-and-white images mapped to the video footage.

Figure 28. An inverted frame of real-time silhouetted video footage.

The first task was to model the IR floodlights used in the *[16]Revolutions* setup illuminating the cyclorama in the installation environment. This artist originally purchased two 36-LED IR illuminators for this purpose, but upon initial testing found that these lights were not diffuse or strong enough to evenly light a wall area of 7.5’x10’, a predetermined area that would be adequate for the cyclorama in the installation. Multiple stronger lights, such as floodlights, would be needed to evenly light this backdrop.

In order to create more powerful lights, this artist researched filters that could be placed in front of strong light sources to isolate the IR wavelength. Although there are professional IR filters available, this artist found most of them to be rather expensive and too small to accommodate floodlights. Electronics enthusiast Don Klipstein offered a more cost-effective solution on his website using a combination of stage lighting filter gels to block visible light and pass IR. Gels subtract wavelengths of color from visible light to achieve desired effects. He observed that an IR pass filter could be fashioned using the following combination of Roscolux (maufactured by Rosco Laboratories Inc.) theatrical lighting gels [34]:

- 2 - #19 (“fire” – strong red amber),
- 1 - #83 (medium blue), and
- 1 - #90 (dark yellow green).

This particular combination of filters can be placed in front of a household floodlight to extract and pass solely the IR wavelength, resulting in an IR illuminator much more powerful and diffuse than the 36-LED ones used in the initial testing setup. Gel sheets are available at theater production supply stores and, at the time of this writing, cost under $10 USD each, offering an inexpensive illumination method.

3.1.3 Visual Considerations

Inspiration for the style of the piece is drawn from the artistic techniques of video artist Bob Sabiston, the work of painter Jackson Pollock, and digital artist Camille Utterback.

The painterly rotoscoping techniques of Bob Sabiston [35] have refocused attention to hand-drawn media as a form of visual communication. In this context, *rotoscoping* is the process of superimposing drawings over video frames. The final product yields a unique visual quality that is unattainable by traditional 2D or 3D techniques: a generalization of color and an emphasis on line weight and quality. Sabiston's visual style suggests a reality not of this physical world, but of an alternate one while maintaining a clearly representational image (Figure 29).

Artists Jackson Pollock and Camille Utterback are points of inspiration because of their evident appreciation for mark-making and interactivity. In their work, interaction fosters the creation of marks. Pollock, for example, used gesture to create loose, dynamic, and unrestricted marks. The freeness of the painterly marks in the resulting work ignores fundamental compositional elements and lacks a focal point. This "action painting" technique (Figure 30) reflects "the act of painting as a spontaneous, unprepared gesture unconstrained by the effort to create a representational likeness" [36]. Utterback creates physical-digital systems that engage people's bodies, heightening their perceptions of spatial relationships, gesture, and body language. The environments she creates promote freedom of movement, as participants need only enter in the installation space to begin an interaction. In Utterback's work, movement is essential to participating in (or generating) a visual virtual environment and establishing a channel of communication between the real and virtual worlds.

Figure 29. A frame from Sabiston's *A Scanner Darkly*, 2007 [35].

The stark contrast of the artificiality of digital media versus the materiality of paint that Pollock explored provides an interesting contrast between the physical and virtual worlds. One goal of the installation was to extend depth (three-dimensionality) to the painting medium that Pollock was unable to explore, with specific attention to the loose gestures characterized by his work. In addition, the experience this artist wanted to facilitate was the ability for anyone to be a part of the art making process, just as Utterback so successfully accomplishes with her installation spaces.

Figure 30. Pollock's "action painting" style.

3.2 Phase 2: Creating the Physical Installation Space

3.2.1 The Cyclorama

In order to achieve effective IR lighting in the installation setting, this artist set to model an environment similar to *[16]Revolutions*. An old collapsible 7.5'x10' Da-Lite® industrial metal frame projection screen was obtained from a theater surplus source (Figure 31). This screen would serve as a durable cyclorama: it was made of white reinforced pliable plastic that could be stretched and adhered to a frame using snap fasteners.

Figure 31. The 7.5’ x 10’ cyclorama stretched out over its frame.

3.2.2 The Lighting Rig and Setup

To create the IR illuminators, this artist obtained Roscolux gels (see section 3.1.2.4) to construct an IR-pass filter for four clamp work lamps. Each gel sheet was cut into fourths to accommodate this need. The gels were then layered according to the specified combination and affixed to the work lamps using clothespins. Each of these work lamps was equipped with a 150-watt floodlight for maximum light dispersion. The gels are specially designed to withstand the high heat emission of stage lighting, provided that they do not touch a light’s surface. In one of the initial lighting tests, this artist accidentally melted a set of filters because they came in contact with the floodlight’s surface. Consequently, the gels were affixed to an aluminum foil extension fashioned for each of the work lamps (Figure 32).

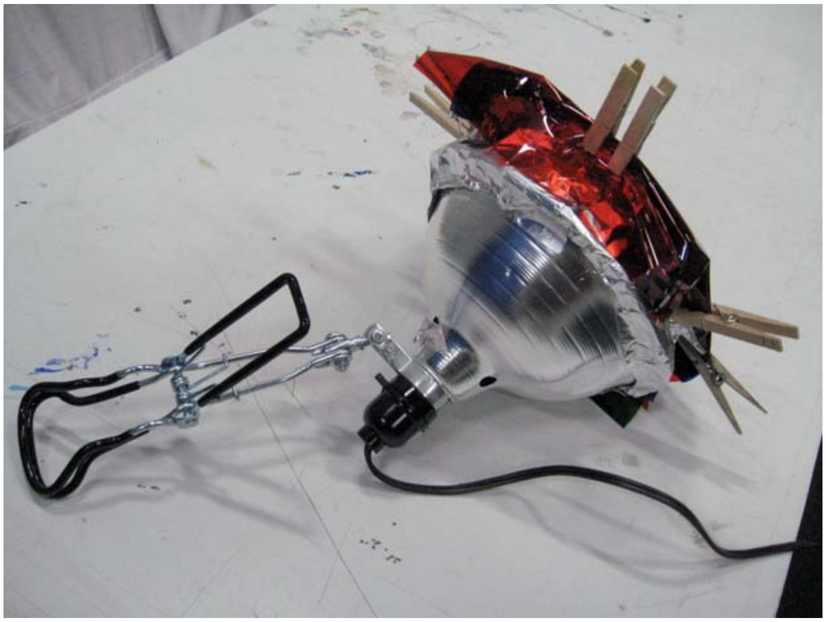


Figure 32. A clamp work lamp equipped with a flood light and lighting gels.

The IR illuminators needed to light the cyclorama evenly, so this artist created two simple lighting rigs that would stand on either side of the cyclorama. These rigs were inexpensive: each one was constructed using particleboard and a PVC pipe (10 ft.), connector, and threaded flange. Screwing the flange to the particleboard created a base that enabled the pipe to stand when attached with a screw connector (Figure 33). Once the lighting rigs were constructed (Figure 34), the IR illuminators could clamp easily to the pipes and be directed towards the cyclorama (Figure 35). This configuration made

Figure 33. Attaching the flange to the particleboard for the lighting rig base.

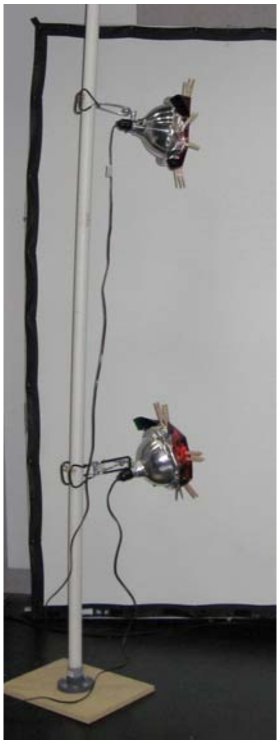


Figure 34. A completed lighting rig.

the lighting setup both portable and easy to adjust. To keep light concentrated on the cyclorama (and for aesthetics), this artist used a few cardboard presentation boards to enclose the lighting rigs and reflect light on to the cyclorama. This consideration helped to create a soft glow of even IR light on the cyclorama that was favorable for the installation.

Figure 35. The lighting setup with the cyclorama.

3.2.3 The Camera Setup

Another important feature of the installation space is the camera setup, which consists of an IR camera and a standard web camera. The camera feeds are integral to the interactive, spatial, and immersive aspects of the installation.

3.2.3.1 The IR Camera

For the installation, this artist used a day/night deView® integrated IR bullet camera equipped with a light sensor to determine the camera mode. Low light levels trigger the camera to enter "IR mode", which activates 18 built-in IR LEDs capable of illuminating an area of about 40 feet from the camera. This mode enables the camera to detect subjects in dark settings, providing this data in the form of grayscale footage. Ample light levels trigger the camera to enter "standard color video mode", in which the camera functions as a typical color video camera. The IR LEDs are not activated in this mode, as ambient light sources are ample enough to accommodate a standard color video feed.

The camera needed to operate slightly differently, however, in order to work within the installation environment. It needed to detect only IR light reflected off of the cyclorama. This type of functionality would facilitate real-time masking (as outlined in 3.1.2.4 and Figure 28). Accordingly, the camera was specially rigged to fit this need (Figure 36):

- The light sensor was covered to "trick" it into constant "IR mode", making the camera only sensitive to IR light regardless of lighting levels,
- All 18 built-in IR LEDs were fully concealed with gaffer's tape because the only source of IR light that this artist needed the camera to read was that reflected by the cyclorama, and

- A special IR-pass filter (see section 3.1.2.4) was constructed to place in front of the camera lens to ensure that only IR light was processed by the camera.

The IR video feed from this camera is analog and requires the use of a converter device to reinterpret the feed as digital video. This digitizing process makes it possible for the computer to utilize and filter the IR information from the installation environment.

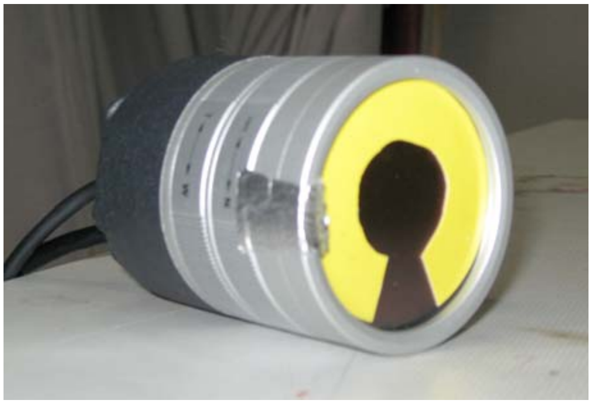


Figure 36. The completely modified and ready-to-use IR camera.

3.2.3.2 The Standard Web Camera and Camera Setup

In addition to the IR camera, a web camera was used in the installation environment. This artist opted to use a USB-powered Apple iSight camera, a compact web camera that produced good quality video. The IR camera and iSight were fastened together (see Figure 37) and their views were calibrated to match as closely as possible. This camera setup was then oriented so that only the cyclorama was in each camera's field of view. With the calibration and camera orientation complete, the IR feed could subsequently be used as a mask to help extract a subject in the installation environment from the web camera feed in real-time. This idea is illustrated in the next section in Figure 39.

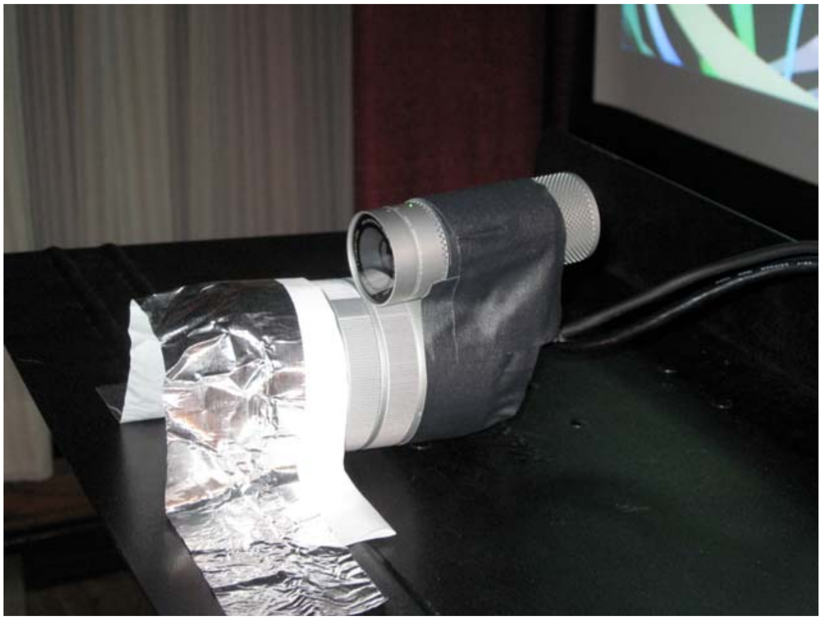


Figure 37. A setup of the IR and color video cameras.

4. IMPLEMENTATION

With the elements of the physical space constructed—screen, lighting, and cameras—this artist was ready to realize the installation space and integrate the virtual elements to bring the worlds together. These virtual elements included writing and utilizing computer programs to process sensor and video inputs to achieve desired visual results. The final phases of the installation's execution were contingent upon a careful balance of resources, requiring iterative planning and construction, assistance from experts in related fields, and continuous experimentation.

4.1 Phase 3: The Layout of the Installation Space

The size of the installation space and the orientation of its components were crucial aspects to understanding how the virtual elements could integrate with the setup. This artist needed to determine the workable dimensions of the installation given the limitations of the cyclorama's size, the range of functionality of the wiimote based on distance from the sensor bar, and the field of view of both video cameras. In addition, the method of viewing the final visual product within the installation space needed to be considered.

Previous measurements and testing revealed the following information:

- The size of the cyclorama screen is 7.5'x10', and
- The most effective working range of the wiimote with respect to the sensor bar is within a distance of about 10 feet.

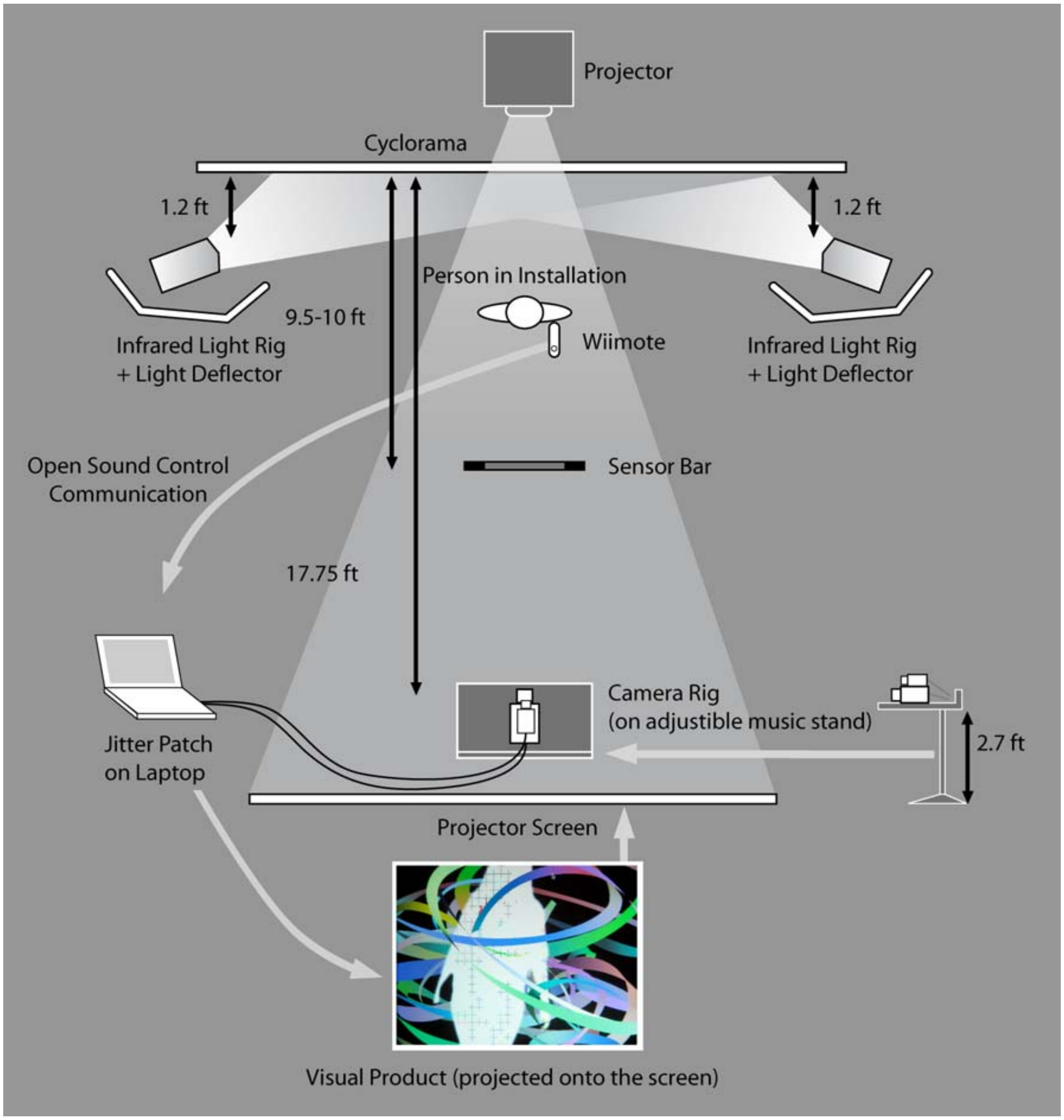


Figure 38. The schematic of the installation space.

Using this information as a starting point, this artist began to draw up a schematic for the layout of the installation (Figure 38). The cyclorama and sensor bar were positioned facing each other with a distance of about 10 feet between them. The sensor bar was placed at a height between 2½ and 3 feet to accommodate wiimote receptivity.

The lighting rigs were placed on either end of the cyclorama, aimed about 45-degrees to light the screen as evenly as possible. Science fair display boards were used to help concentrate and reflect all light from the rigs onto the screen—an addition that helped to improve the contrast of the IR video feed. The cameras were oriented toward the cyclorama and placed at a distance such that the greatest surface area of the screen could fully occupy their fields of view. Testing the camera setup at various lengths disclosed the optimal distance to be about 17 feet 9 inches from the cyclorama and at a height of about 3 feet. This setup allowed for freedom of movement within an area of roughly 55 square feet.

A wall or other type of projection surface was needed to display the final visual product, and was positioned opposite the cyclorama and behind the cameras. This orientation was desirable because it would enable the user to stay situated in front of the cyclorama while still being able to view the final product as s/he interacted with the installation environment. A projector would be used to display the final visual product on the projection surface as the computer processed the video information from the installation environment.

The evaluation of the space provided valuable information for visualizing all physical aspects of its construction. The virtual elements—all computer-based computations facilitating the appearance of the final visual product based on actions of the participant in the space—could now be developed, tested, and easily integrated into this setup.

4.2 Phase 4: Generating the Final Image

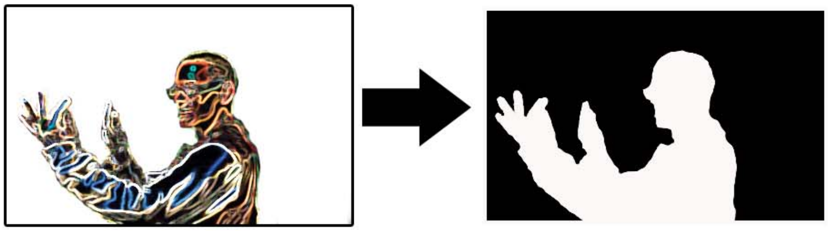
The final visual product, in the form of processed video from the installation environment, is created using a multi-step process. Discovering an efficient procedure to obtain desired results required pre-visualization, iterative testing, continual learning, creative problem solving, and communication with other intermedia artists.

4.2.1 Outlining the Process

The process of generating the final image is comprised of four major steps (illustrated in Figure 39). As previously stated, two cameras—a standard web camera and an IR camera—are calibrated such that their fields of view are as exact as possible. Images of the user against the cyclorama from the standard web camera are obtained and undergo a small series of filtering operations, giving them a slightly abstract appearance (Step 1). The mask generated using the IR camera is used to extract the user from the environment in real-time (Step 2). The user can produce motion and gesture data using the wiimote to create an original 3-dimensional (3D) artistic piece in a painterly style as a form of "virtual" action painting (Step 3).

The extracted images of the user (the combination of Steps 1 and 2) are composited into the artistic environment (of Step 3) as still frames that are updated multiple times per second—enough to give the perception of real-time interactivity. This gives the impression of enveloping the user in a 3D world that s/he has created (Step 4). This experience is made unique by enabling the user to actually "wade" through the 3D space of his/her piece, as if walking through and exploring his/her own painting. This part of

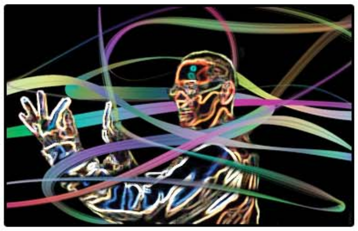
Visual Considerations



Step 1. Image of participant in scene, filtered using a combination of real-time filters (cyclorama in background)

Step 2. Image gathered from IR camera using IR-light lit cyclorama

Step 3. Image created using wiimote
- IR receiver xy location determines how strokes are laid on virtual canvas



Step 4. Final generated image based on participant's z-positioning in the installation space

Figure 39. Visual considerations and process for generating the final composite image.

the immersive experience is generated based on the user's z-movement (distance from the cameras) within the installation space. The participant within the installation space views this resulting image.

4.2.2 Writing the Program to Bridge the Physical and Virtual Worlds

This artist had a working idea of many of the virtual elements of the installation once the process had been pre-visualized. The bridging of the physical and virtual worlds now needed to be facilitated using programs that could accomplish the tasks outlined in this planning process.

4.2.2.1 Capabilities of the Program

The required capabilities of the main program, based on detailed assessments of the pre-visualization process, were determined by this artist to be the following:

- Sensor Input and Manipulation
    - Obtaining gesture and event data from the wiimote,
    - Channeling this data to Max/MSP/Jitter,
    - Using this data to inform virtual line-drawing, and
    - Storing z-positions of the points making up the line (as they are generated) based on the participant's location within the installation space at any particular time.
- Video Input and Manipulation
    - Reading in camera input from the IR and web camera,

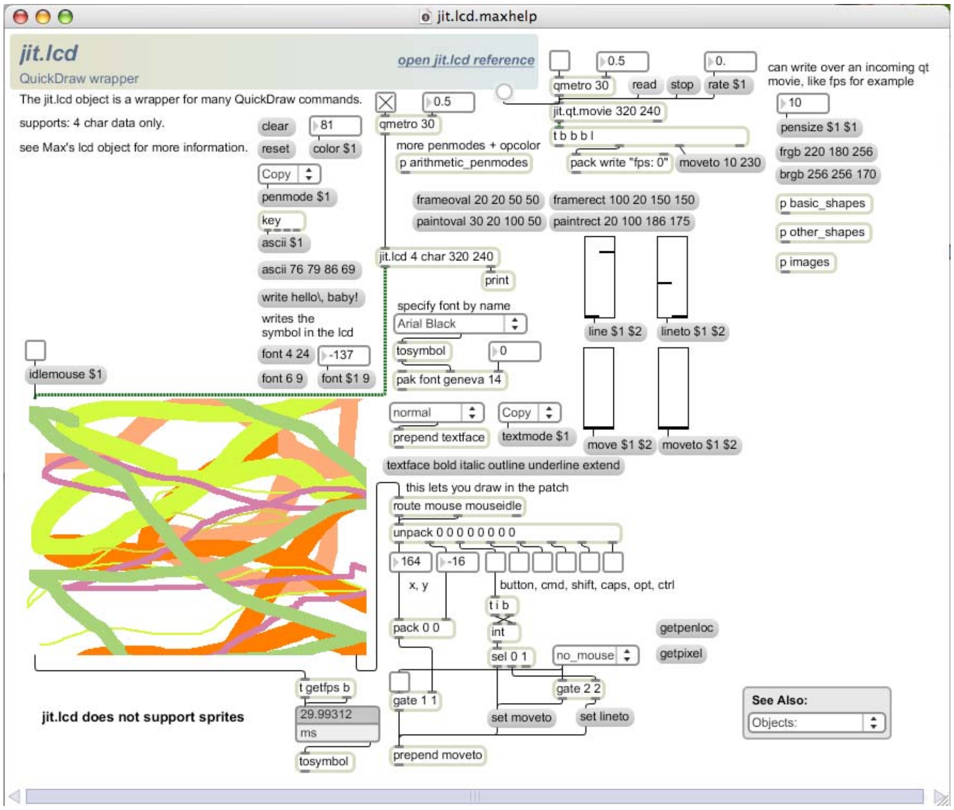
    - Processing, filtering, and compositing the two video feeds together in real-time,
    - Mapping the resulting feed onto a video plane and enabling z-axis movement of this plane in 3D space, and
    - Driving z-axis movement of the video plane based on the participant's location within the installation space.
- Video and Sensor Input Integration
    - Compositing the video plane with the line to achieve the desired visual product; z-axis movement of the plane and stored z-position of the line points based on participant location in the installation space create the illusion of depth.

Each of these facets had their own development challenges, so this artist approached these needed capabilities in sections. The final program was constructed in two main parts—video and sensor input/manipulation—then combined together.

4.2.2.2 Sensor Input and Manipulation

The mark-making based portion of the program was written first because this artist perceived it as the most challenging virtual aspect of the installation space. The method of mark-making went through multiple iterations. A few trials were faced during this process, notably efficiency problems, which required some assistance from outside sources.

Mouse-based drawing was used for initial testing purposes before replacing it with wiimote-driven control with OSCulator (see section 3.1.2.3). Preliminary investigation was done using Jitter's jit.lcd object (Figure 40). This object serves as a wrapper for QuickDraw (a depreciated Macintosh-based graphics library) commands and was considered for its simplicity and efficiency. Lines were easy to create within the jit.lcd window using cursor information, and felt much like digital painting in Adobe Photoshop. However, the capabilities of this 2D-based graphics library were limited in terms of line quality and ability to be composited with live video. Although the stroke size and color of the line could be controlled as it was drawn in the jit.lcd window, other aspects such as color variation and texturing of the line were not possible. This artist felt that the inability to attain these qualities resulted in a lack of visual appeal and dimensionality. Furthermore, a z-depth value could not be applied to this 2D line and the output of the jit.lcd window could not be externalized to other processes, inhibiting the ability to composite it with video in 3D.


jit.lcd

QuickDraw wrapper

The jit.lcd object is a wrapper for many QuickDraw commands.

supports: 4 char data only.

see Max's lcd object for more information.

can write over an incoming qt movie, like fps for example

more penmodes + opcolor

writes the symbol in the lcd

specify font by name

this lets you draw in the patch

x, y

button, cmd, shift, caps, opt, ctrl

jit.lcd does not support sprites

Figure 40. The jit.lcd object.

The OpenGL standard for drawing is capable of texturing lines, providing dimensionality with the use of its 3D libraries, and compositing multiple objects in a 3D environment. Consequently, this artist searched through Jitter documentation and the Cycling '74 (the manufacturers of Max/MSP/Jitter) forums online to find ideas for an OpenGL-based method of line creation. A simple OpenGL-based paint program in Jitter was located on the forums and used as a testing platform for development (Figure 41). A brush tip collection, supplied with the patch, simulated various types of brush strokes, much like the paintbrush tool in Adobe Photoshop. Although the appearance of the strokes was fluid, much like watercolor, it began to take on a "cookie cutter"-type feel because the texture was clearly repeated over and over to create the stroke. This characteristic became more apparent when the mouse cursor moved too fast for the refresh rate of the jit.lcd window, resulting in spotty line breaks. Additionally, because multiple points were needed to create a line, simultaneous tracking of all of their z-positions would be required in order to give it a realistic illusion of depth when integrated with video. This would necessitate increased computer processing power as the line grew in size.

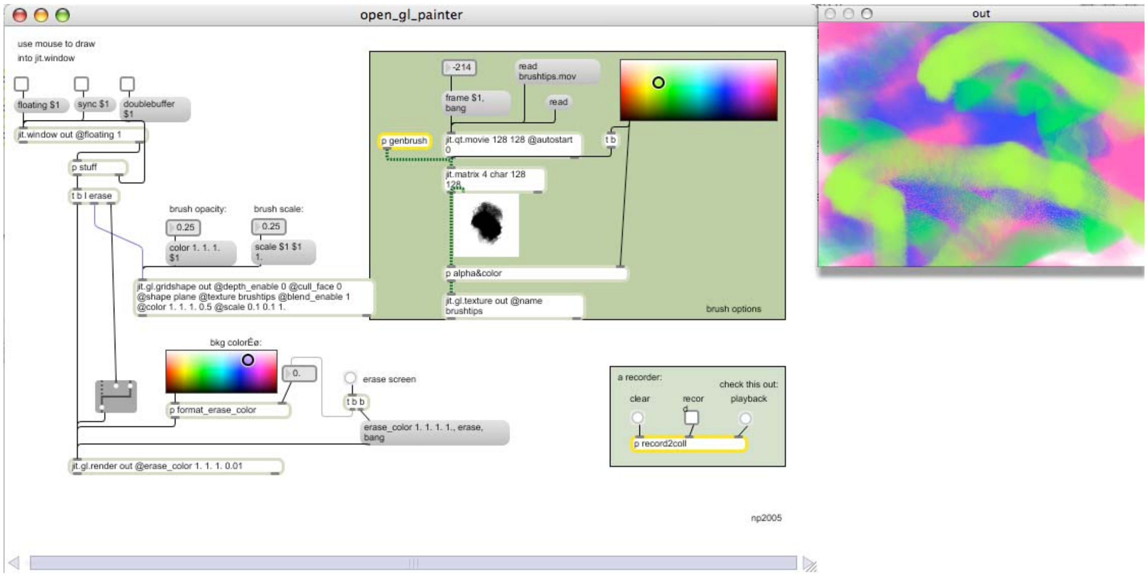


Figure 41. A simple OpenGL paint program obtained from the Cycling ’74 forums.

The next method of OpenGL-based line creation this artist explored involved anti-aliased spline curves, a method of line generation that interpolates between points (known as “control points”) to generate a series of smooth curves (see Figure 42). This technique produces flowing line results and requires significantly fewer points than the previous approach to guide its generation. A sample patch included in the Max 5 documentation called jsui_splinestuff (Figure 43) contained some of the basic ideas for generating this type of line in Jitter using jsui ("JavaScript user interface"), an object driven by the JavaScript programming language. Spline curves were generated in the jsui window within the patch using a user-defined number of control points (a default of eight), each with randomly generated x, y, and z values. These random curves could be fine-tuned by modifying values in the patch user interface to affect variables within the script that would influence attributes such as line color, scale, rotation, and texture. Although this program had other interesting capabilities, such as a wireframe mode to view the control points and a morphing feature to blend between previous and current iterations of generated curves, they were not of specific relevance to this thesis. The unique ability of JavaScript files loaded by jsui to have access to the OpenGL API—a feature normally extrinsic to the JavaScript language—was of particular interest to this artist. This added capability enabled the creation of lines completely within the script.

Although the visual results attainable by the jsui object were appealing to this artist, research on the Cycling '74 forums revealed that it was notorious for being CPU-intensive and inefficient. The jsui object uses software rendering to generate graphics—that is, rendering takes place completely in the CPU. This object could have a detrimental effect on a Jitter patch if it occupied too much memory, as it would slow down other processes. An alternative to using the jsui object that was presented in the Cycling '74 forums was to use Jitter's jit.gl.sketch object to obtain the same visual quality with hardware rendering instead of software rendering. Hardware rendering

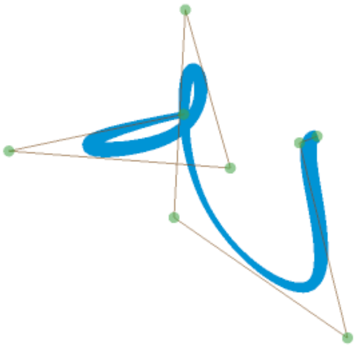


Figure 42. A spline curve generated from eight randomly-generated curve points.

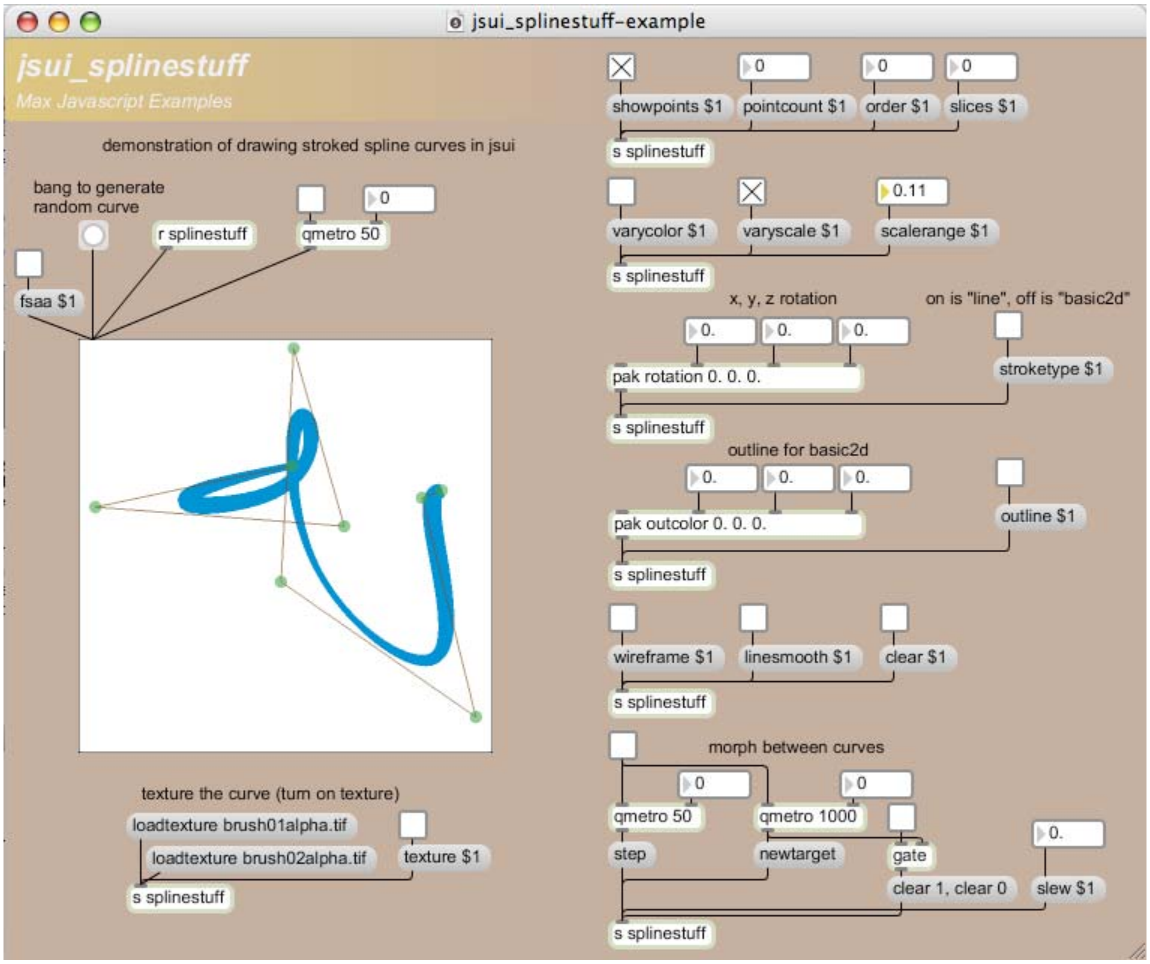


Figure 43. The jsui_splinestuff patch.

takes place entirely within a computer's graphics card, a committed graphics rendering device also known as a graphics processing unit, or GPU. These devices are more efficient than CPUs in terms of computing power and their ability to manipulate and display graphics. The jit.gl.sketch object records a list of 3D drawing commands that include a majority of the OpenGL API.

The most helpful resource for translating the basic framework of jsui_splinestuff into a more efficient patch (without the jsui object) was a tutorial in the Jitter documentation regarding object callbacks [37]. This tutorial used a JavaScript file to draw an OpenGL scene that was made interactive through mouse events using a listening and callback mechanism known as a JitterListener (Figure 44). This drawing context was displayed in a separate window (a jit.window object) whose size could be made full screen—a useful feature that could be utilized when projecting the computer's visual output within this artist's installation space. The tutorial patch created a sphere to track the mouse cursor's position in the window. This sphere could influence the position of smaller spheres in the window by colliding with them.

Figure 44. The JitterListener tutorial patch.

The documentation in this tutorial notes that:

> An entire OpenGL Jitter patch can be encapsulated in JavaScript by instantiating jit.gl.render, jit.window, and other OpenGL objects within procedural code written for the js object. These objects have all their messages and attributes exposed as corresponding methods and properties. You can use a JitterListener object to respond to events triggered by a Jitter object within JavaScript. The JitterListener then executes a callback function, passing the calling message to it as its argument. This allows you to write JavaScript functions to respond to mouse interactivity in a jit.window object, file reading in a jit.qt.movie object, and other situations where you would want to respond to an event triggered by a message sent by a Jitter object out its status outlet in a Max patcher. [37]

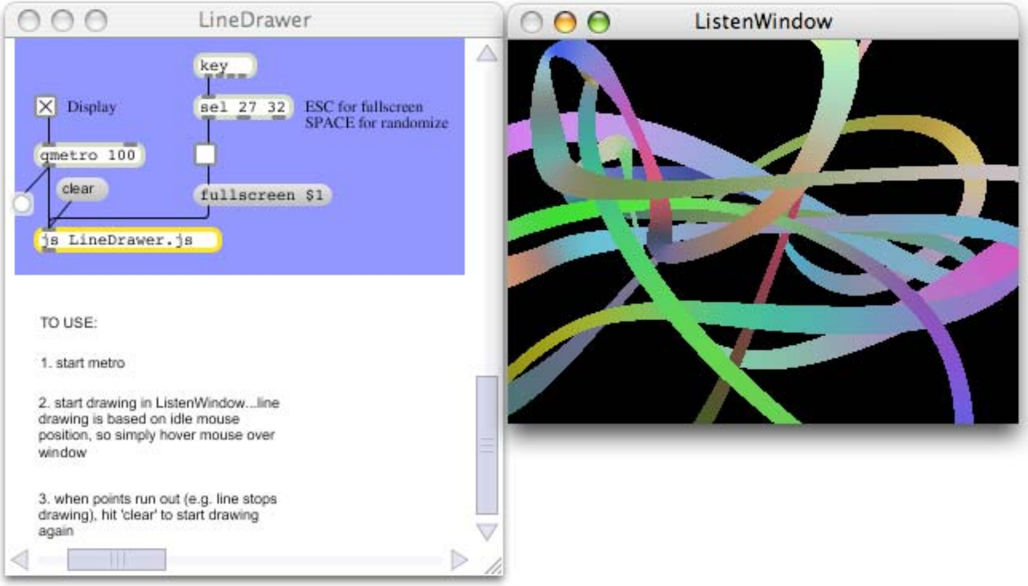
Given this information, this artist began to develop a more efficient, hardware-based patch to draw a line established by mouse movement. This patch would be constructed using Max/MSP/Jitter's .js, JitterListener, jit.gl.sketch, jit.gl.render, and jit.window objects and avoid the use of jsui. Like the tutorial patch, this artist wanted mouse position to be monitored using a JitterListener object. These positions could then be stored and used to generate a line.

The final patch for this iteration of development is shown in Figure 45. The js object, as stated earlier, contains JavaScript code that can encapsulate an entire OpenGL Jitter patch. The code needed to be able to draw the OpenGL scene to a window and handle interactive events from the mouse (such as obtaining its x-y position within the window). This artist created five main entities in the global block of the JavaScript code to facilitate this:

1. jit.window object: displays the drawing context,
2. jit.gl.render object: performs rendering of the OpenGL objects (associated with jit.window only) in the jit.wiindow,
3. jit.gl.sketch object: records a list of OpenGL drawing commands,

4. JitterListener: monitors (“listens to”) jit.window and calls a callback function to handle events when triggered by them, and
5. An array to hold a finite number of (x, y, z) coordinates corresponding to control points making up the line.

The JitterListener object constantly monitored and handled the x-y mouse position within the output screen. For example, position was only recorded and placed in the control point array while the left mouse button was pressed. Up to 200 control points could be stored in this array to generate a line before requiring the scene to be cleared (using a “clear” button provided in the patch). The depth of these control points was not being considered yet, so their z-positions were ignored and set to 0. As the position values populated the control point array, the line was rendered to the screen. The scale and coloring of the line were coded in the script to be set at random values as the line was generated. A qmetro object was used to prompt the main drawing loop in the script at regular intervals to check for mouse events that would cause the visual product on the screen to update. Figure 45 reflects a qmetro prompting interval of 100 milliseconds, or every 1/10th of a second. The final scene was rendered using an orthographic projection, meaning that the “z-axis of the drawing context was ignored in terms of sizing objects based on their distance from the virtual camera” [37].



TO USE:

1. start metro

2. start drawing in ListenWindow...line drawing is based on idle mouse position, so simply hover mouse over window

3. when points run out (e.g. line stops drawing), hit 'clear' to start drawing again

Figure 45. A hardware-based patch for OpenGL line drawing
based on mouse movement.

Surprisingly, the patch seemed to perform very poorly once it was completed. The line was generated smoothly at its inception, but as points were added, the generation of the line became very sluggish. Something was weighing down calculations over time to use up to 100% of the CPU. This artist turned to the Cycling '74 forums for assistance after considerable time was spent troubleshooting this problem to no avail [38]. Upon submitting this issue to the Jitter forum, this artist received a response from Jitter developer Joshua Kit Clayton. The jit.gl.sketch object in my patch was building up an internal command list that was not reset before each iteration of the main drawing loop. This caused this command list to occupy increasing memory as it grew in size. This fix was simple, requiring only one additional line of code. It made a significant difference in the performance of the patch: about 25-30% CPU resources were now being used versus the 100% noted before the problem was addressed. Expectedly, using this hardware-based method was more efficient, cutting the CPU usage by at least half that of the resources needed to run the jsui_splinestuff patch (approximately 60% CPU).

With the successful development of a more efficient line drawing patch, this artist looked to replace mouse cursor position listening with the monitoring of x-y pointing coordinates of the wiimote (see section 3.1.2.2). OSCulator, described in section 3.1.2.3, facilitates the transfer and interpretation of wiimote information from the controller to the computer via a Bluetooth connection. A sample Max patch (included with OSCulator 2.5.6; Figure 46) illustrates the connections needed to use this object to access OSCulator's information—particularly the x-y pointing coordinates and button events of the wiimote—in the programming environment. This information is channeled to Max/MSP/Jitter using a port for communication (the sample patch uses the default port 9000) and the osc-route object. This object, developed for Max and made available online as part of the CNMAT extension by the University of California at Berkeley Center for New Music and Audio Technologies, provides message dispatching through an OpenSoundControl address space [39].

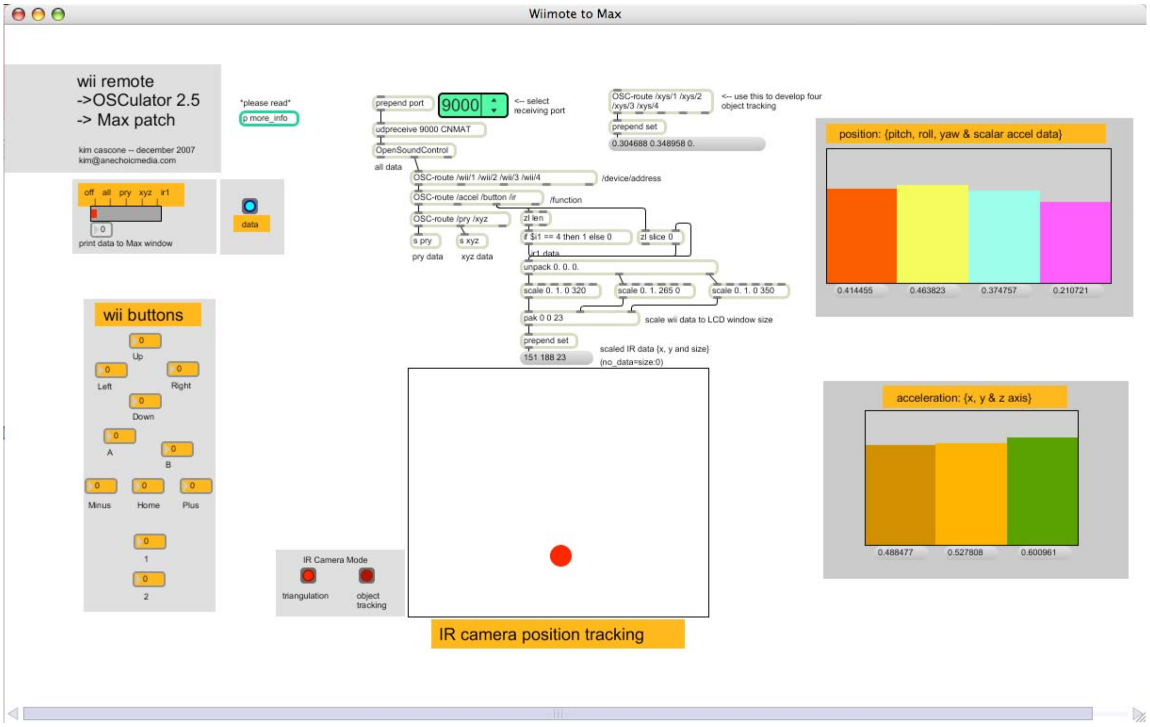


Figure 46. A sample Max patch for establishing connectivity with OSCulator.

Like the left mouse button used to trigger drawing in the previous patch iteration, this artist wanted a button to be pressed on the wiimote in order to draw a line. The callback function used in the previous code for mouse events was replaced with a different function to monitor the events of the “B”, or trigger, button on the wiimote. If the trigger button was pressed down, the script would know to record and place the current calculated x-y pointing coordinates of the wiimote into the control point array. Additionally, the “A” button was programmed to clear the screen (and thus the control point array) when pressed.

The x-y pointing coordinates supplied by OSCulator are screen coordinates that are derived and approximated when the wiimote and sensor bar are used in tandem (see section 3.1.2.2). This artist needed to convert these pointing values from screen to world coordinates so that the line could be drawn correctly to the output window. Conversion between these systems was performed by normalizing coordinates to the range of -1 to 1 for y values (bottom to top) and –aspect to aspect for x values (left to right), where "aspect" is the ratio of width/height of the output window. It is these normalized coordinates that are truly placed in the control point array. The main drawing loop, prompted by the qmetro object, uses the control point array to draw the line to the screen. The z-depth of the control points making up the line during this iteration of patch development was still ignored; this artist would add functionality for it once the video portion of the patch was complete.

The end of this iteration resulted in a working patch, more efficient than jsui, that could generate a line in OpenGL based on events and information from the wiimote. The generative quality of the resulting marks, using the wireless wiimote as a digital paintbrush, reflected a spirit of freeness found in the loose gestures of Jackson Pollock's action painting. The capability for these lines to have z-depth taps in to an added element of dimensionality that Pollock, however, was unable to explore.

4.2.2.3 Video Input and Manipulation

With perhaps the most challenging aspect of the program development complete, this artist moved on to the video processing aspects. The video streams from the IR camera and web camera were read into Max/MSP/Jitter using the jit.qt.grab object, which digitizes video from an external source and prepares it for use with other Jitter objects. Once in Max/MSP/Jitter, the video data is sent through a series of processes, outlined in Figure 47.

The web camera video is calibrated with the IR camera video such that both fields of view are as exact as possible. This is done by modifying and interpolating the viewing dimensions of the web camera video feed. A combination of Jitter filters is used to process the video, including jit.sobel, jit.tiffany, and jit.hatch (Figure 48). Each video filter produces unique visual results based on a set of object-specific parameters. These filters were chosen for their interesting visual characteristics, including color generalization and line quality (as in Bob Sabiston's work, section 3.1.3), and are combined mathematically to obtain visual results unattainable with a single filter (Figure 47, A).

The IR camera video is converted to monochrome using the jit.rgb2luma operation and tweaked using a math operation to remove any extraneous noise from the video that might result from slightly uneven lighting of the cyclorama. When these masks are used, white areas denote full opacity and black areas denote full transparency. Two masks are created from this video, each the inverse of the other. The first mask (Figure 47, Mask 1) is slightly blurred using jit.fastblur to smooth any anti-aliasing caused by this processing. This monochromatic video can then be used as a mask for the web camera video (Figure 47, B). The second IR video mask (Figure 47, Mask 2) is used in two separate processes. In the first process, it is slightly blurred like Mask 1 to smooth anti-aliasing (Figure 47, C). The second process will be outlined in the next section.

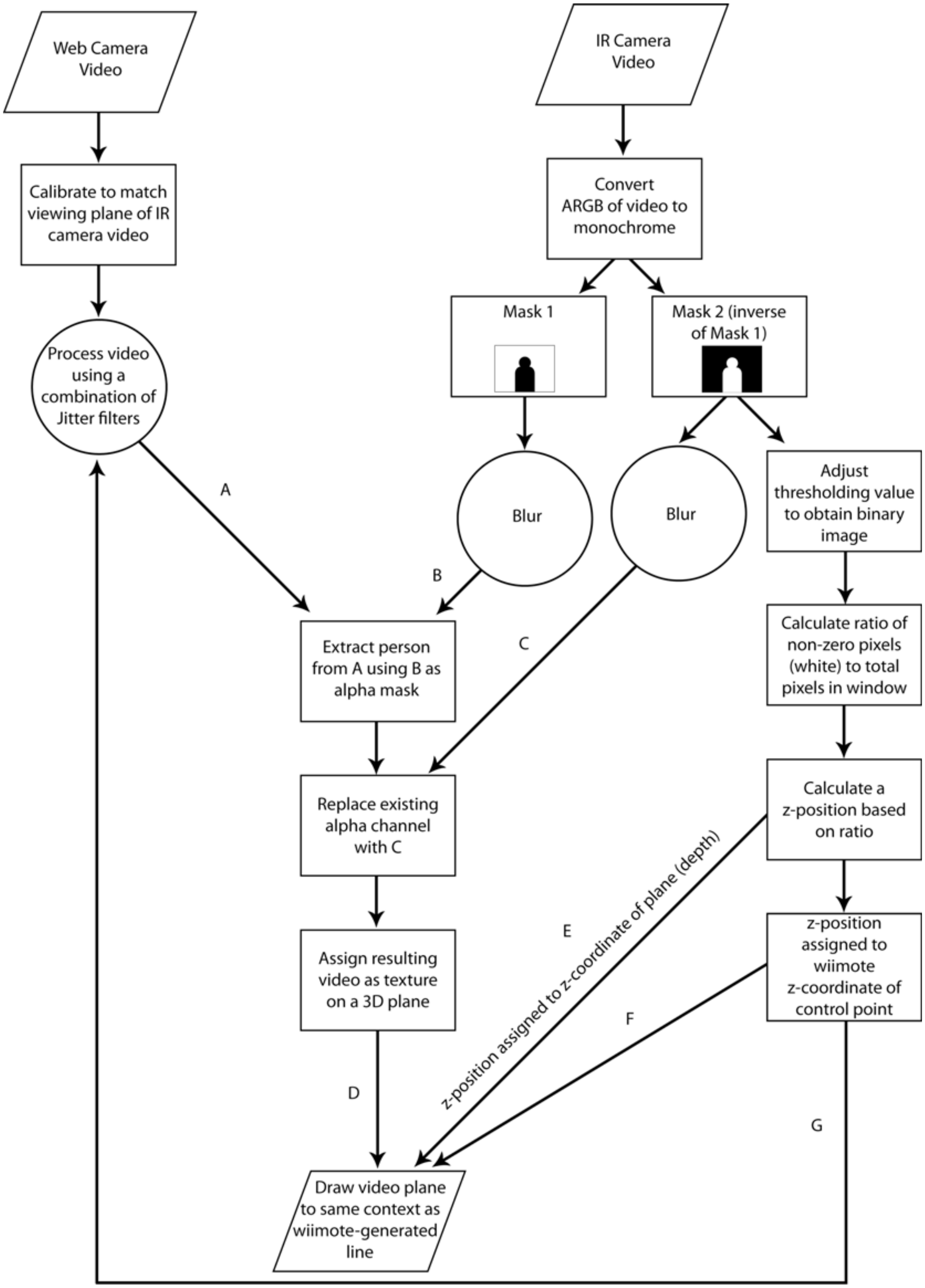
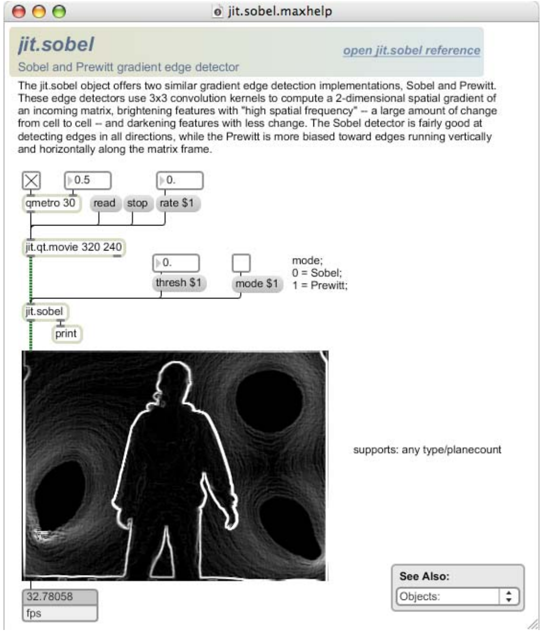


Figure 47. Video manipulation processes in Max/MSP/Jitter.



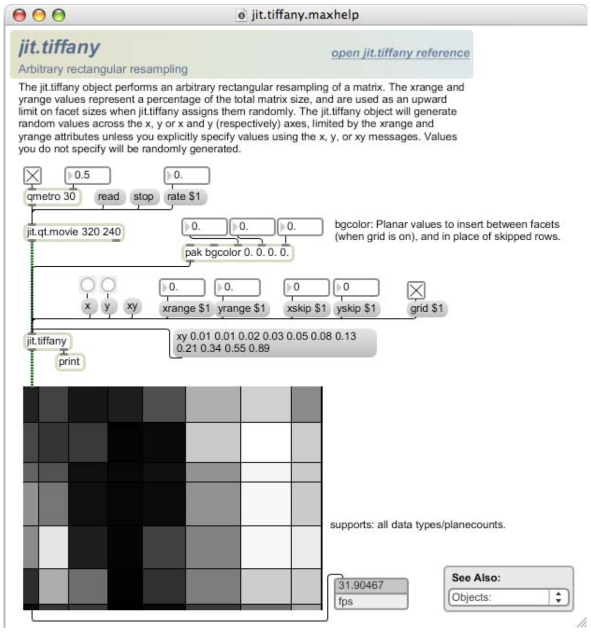
jit.sobel

Sobel and Prewitt gradient edge detector

The jit.sobel object offers two similar gradient edge detection implementations, Sobel and Prewitt. These edge detectors use 3x3 convolution kernels to compute a 2-dimensional spatial gradient of an incoming matrix, brightening features with "high spatial frequency" -- a large amount of change from cell to cell -- and darkening features with less change. The Sobel detector is fairly good at detecting edges in all directions, while the Prewitt is more biased toward edges running vertically and horizontally along the matrix frame.

mode;
0 = Sobel;
1 = Prewitt;

supports: any type/planecount



jit.tiffany

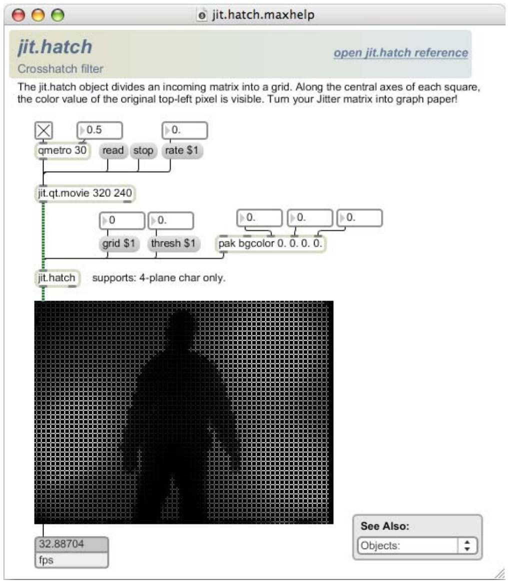
Arbitrary rectangular resampling

The jit.tiffany object performs an arbitrary rectangular resampling of a matrix. The xrange and yrange values represent a percentage of the total matrix size, and are used as an upward limit on facet sizes when jit.tiffany assigns them randomly. The jit.tiffany object will generate random values across the x, y or x and y (respectively) axes, limited by the xrange and yrange attributes unless you explicitly specify values using the x, y, or xy messages. Values you do not specify will be randomly generated.

bgcolor: Planar values to insert between facets (when grid is on), and in place of skipped rows.

xy 0.01 0.01 0.02 0.03 0.05 0.08 0.13 0.21 0.34 0.55 0.89

supports: all data types/planecounts.



jit.hatch

Crosshatch filter

The jit.hatch object divides an incoming matrix into a grid. Along the central axes of each square, the color value of the original top-left pixel is visible. Turn your Jitter matrix into graph paper!

supports: 4-plane char only.

Figure 48. The jit.sobel, jit.tiffany, and jit.hatch filters.

The video data from A and B are combined using the jit.alphamask object, where B is used as the alpha mask to extract a person from A. The alpha channel of the resulting video is replaced with C so that the only portion of video visible in the final product is the person in the installation space. This video is assigned to be a texture on an OpenGL 3D plane. Finally, this video plane is sent to render in the same context as the wiimote-generated line (Figure 47, D). Unlike the difficulties confronted in the development of the line drawing patch, the video-based patch was simpler to implement, but required some careful planning in order to have desirable results. As with the z-coordinate of the control points in the line, the z-depth of the video plane (D) was ignored, but this feature would soon be added once the sensor (wiimote line drawing) patch was integrated into this video patch.

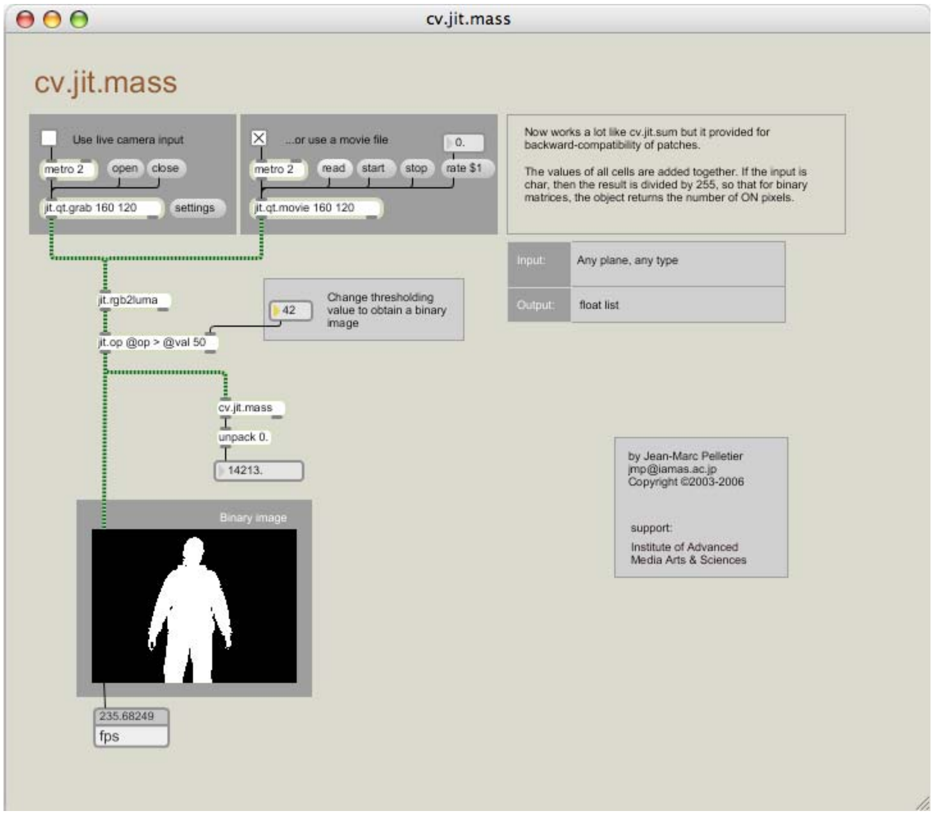
4.2.2.4 Video and Sensor Input Integration

With the essential functionality of the line drawing and video patches complete, this artist added in the final element to bring these programs together and bring immersive quality to the final visual result—the z-depth.

Originally, this artist looked to calculate z-depth with the help of additional sensors or the Raw IR information provided by OSCulator, both addressed in section 3.1.2.2. Additional sensors were ruled out early in the planning process due to concern of overloading the computer processor. Raw IR employs triangulation to approximate distance and seemed to be a viable option because OSCulator was already being used with the line drawing patch. However, upon testing this data in Max/MSP/Jitter, this artist found the data to be too jumpy and unreliable to map to a z-position.

The z-depth needed to be calculated using a low-cost method to minimize overhead on top of the existing processes of the line drawing and video patches. A discussion of this problem with committee member Carol Lafayette revealed the existence of a Jitter object called cv.jit.mass (Figure 49) that could assist in calculating z-positions economically. This object, developed by Jean-Marc Pelletier, is part of the free cv.jit collection of Max/MSP/Jitter tools available online [40].

The previous section (4.2.2.3) outlined the derivation of two masks from the IR camera video. The second IR video mask (Figure 47, Mask 2) is used in two separate processes, the first of which generates C and serves as the alpha channel for the final video. In the second process, an adjusted threshold value is used to obtain a binary (purely black and white) image, preparing the video as it is passed to the cv.jit.mass object. This object is used to return the number of non-zero (white) pixels in a video frame. The more non-zero pixels, the closer a participant is to the camera rig in the installation space. A percentage of non-zero pixels in the frame is calculated and then mapped to a z-position value. This data is funneled to various attributes in the program (refer to Figure 47), including: the z-coordinate of the video plane (E), the z-coordinate of a control point (if a line is currently being drawn with the wiimote; F), and as an input parameter to Jitter filters (G). These additions provided the necessary capability for the participant in the installation space to draw at various z-positions and immerse himself in the visual product depending on his location in the installation space. The Jitter filters also changed in response to the participant's location, affecting aspects such as the hatch size in the jit.hatch filter (smaller for farther z-distances and larger for closer z-distances). All of these elements worked together to encourage the user to understand z-interactivity (the illusion of depth) through visual reinforcement. The final Javascript code is provided in Appendix C. Like Camille Utterback's work (section 3.1.3), the combination of the wiimote drawing and movement-based visuals facilitates the ability for anyone to be a part of (and even personalize) the art making process. With the z-calculation mechanism finally in place, the line drawing and video patches were integrated into one patch.



Now works a lot like cv.jit.sum but it provided for backward-compatibility of patches.

The values of all cells are added together. If the input is char, then the result is divided by 255, so that for binary matrices, the object returns the number of ON pixels.

Input:	Any plane, any type
Output:	float list

Change thresholding value to obtain a binary image

by Jean-Marc Pelletier
jmp@iamas.ac.jp
Copyright ©2003-2006

support:
Institute of Advanced
Media Arts & Sciences

Figure 49. The cv.jit.mass object.

5. RESULTS AND EVALUATION

5.1 Artist Intentions for Final Thesis Work

The final thesis work is intended to provide an opportunity to explore one's personality within a creative boundary. As graffiti is a personal expression in a public space, this artist aims to encourage deliberate (virtual) mark making as a form of personal expression. The participant inhabits a personalized place with his/her image that s/he can manipulate dynamically. Physically, the final video product was created to have evidence of the inspiration of Sabiston's, Pollock's, and Utterback's work and be vibrant and full of color. The styles of the line and filtered video in the final visual product have been designed to work together and complement each other, as they exist in tandem.

By using minimal technology and the widely familiar wiimote as a driving force between the physical and virtual worlds, this artists hopes that the participants in the space will focus on the aesthetic results of the piece and less on technological aspects. It is desired that both artists and non-artists approach the installation environment as an opportunity to leave marks in an alternate reality. The installation facilitates the full immersion, or oneness, of an artist and his/her work by making its creation a fully interactive process. The artist in the installation space should feel as if s/he is an integral part of her/his own creative work because s/he is a part of it and can actually navigate through it. The facilitation of immersion, by virtue of the wireless wiimote, encourages freedom of movement and exploration of articulation.

5.2 Evaluation of Results

Interactive and immersive technology invites us to be participants in another world that we can affect and partake in by using our senses. Creating an interactive, immersive, and customizable mark-making based real-time video installation was a stimulating experience for this artist. Investigating interactive possibilities with digital media facilitated the creation of a piece that offers a refreshing perspective on interactive art. By challenging the boundaries of interactivity as cosmopolitan society perceives it today, this artist hopes to lead those learning about (or experiencing) this thesis work to think more deeply about the interactions we make and have the capability to make.

The primary goal of this thesis was to create a public mark-making art installation with an unobtrusive interface. The resulting installation enabled participants to create personal marks using a familiar gaming controller in an environment that produced real-time interactive visuals to encourage immersion within his/her creations.

Secondly, this artist wanted to promote the idea of isolating and breaking down the dichotomy of user and technology through the use of physical actions for personal interactivity. Participants in the installation area could uniquely affect the visual result by moving about the space and by determining the location and creation of personalized marks. This autonomy in the creative, interactive, and visual processes (through inclusivity and integration into the final product) provides an experience unlike those that can be found exclusively in the physical or virtual worlds.

The next goal was to create an efficient governing system based on real-time processing. The early concepts of the governing system began as a collaborative project with committee member Jeff Morris and his improvisational music trio. This artist created a real time drawing and video-filtering tool (in the form of a Jitter patch) using the wiimote to accompany these performances. The patch went through several iterations over the course of a few months and was presented as an improvisational visual element driven by the wiimote that produced "visual music". With each iteration, this artist learned new ways to think about enhancing the visual quality. This process—echoed in the previous section on sensor input and manipulation— encouraged this artist to find aesthetic and efficient solutions and helped influence the line drawing aspects of the thesis work. Other aspects of the governing system, such as video processing and z-depth calculation, were constructed and tested for efficiency as individually manageable pieces before being integrated together with the wiimote line-generating program.

Vizagogo, a yearly showcase of Visualization Sciences student work at Texas A&M University, offered this artist a valuable opportunity to "test run" the installation (Figures 50 and 51). The work, still under development, was presented to attendees—including families, students, and professors—in a usable form that this artist could evaluate. The feedback regarding the wiimote as a delivery tool and the implementation of the user interface were the most important aspects of study. The audience was comprised of seasoned Wii gamers and older generations not familiar with gaming. Most people found it easy to participate in the installation after a quick operating overview from this artist or by watching others interact with the space beforehand. This artist noticed, as expected, that younger subjects in the installation space had more familiarity with the wiimote device and accordingly knew how to operate it with little guidance. Some of

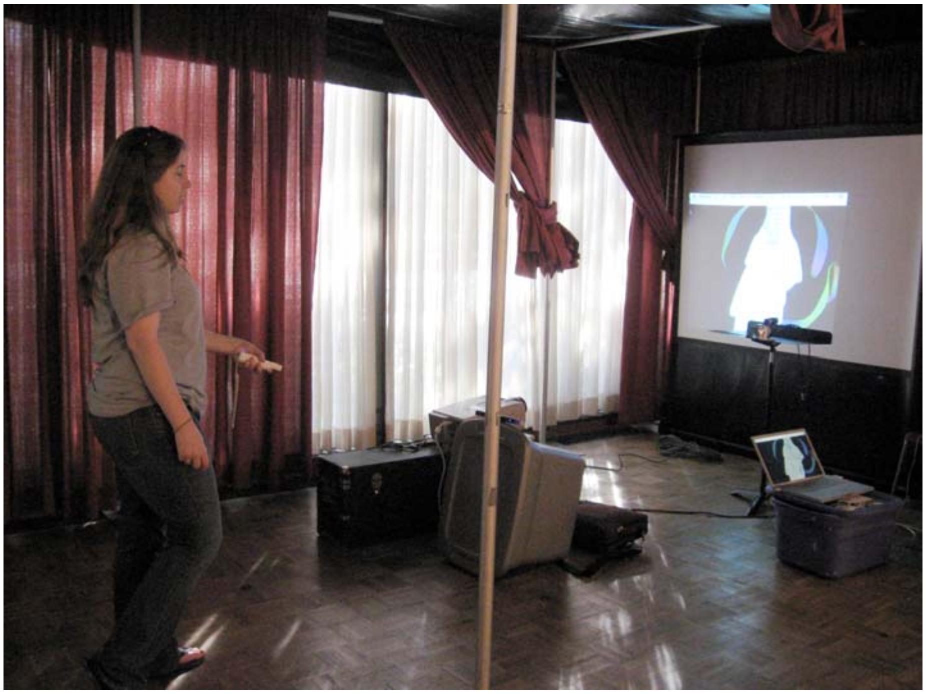


Figure 50. The installation setup at Vizagogo, 2008.

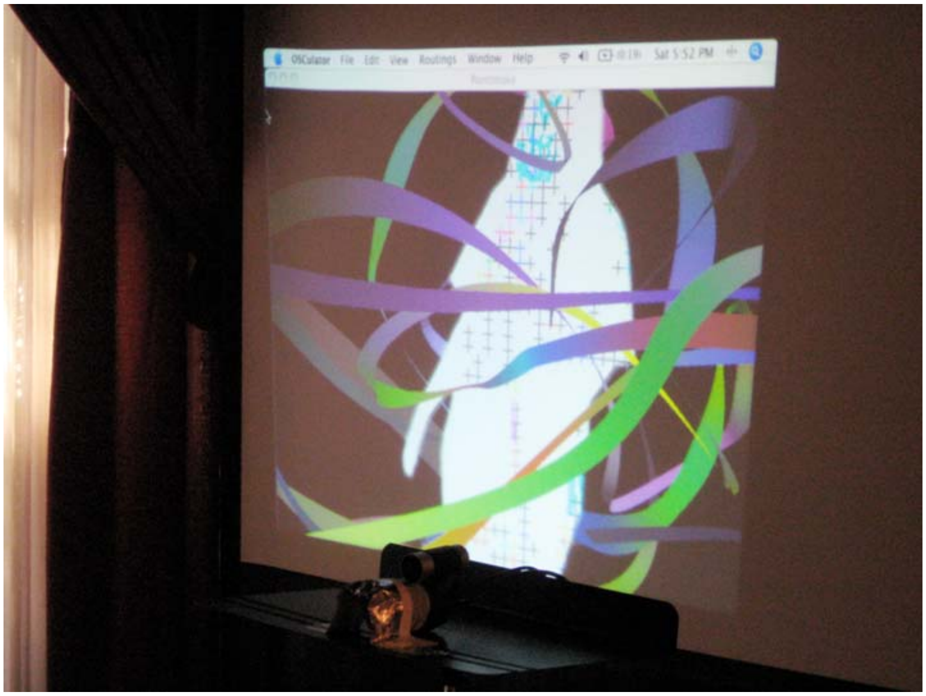


Figure 51. The visual product at Vizagogo, 2008.

the older participants, however, needed more detailed guidance and expected the wiimote to react to large gestural movements—a feature that the wiimote is not capable of due to the limited field of view of its internal IR sensor (Appendix A) [41]. The wiimote IR sensor must detect both of the sensor bar's IR lights within its narrow field of view in order to effectively estimate an x-y screen position (section 3.1.2.2). Large sweeping gestures cause the sensor bar's lights to move out of the wiimote's field of view, and in the case of the installation, can inhibit the generation of the line because x-y positions cannot be tracked successfully. The artist confronted this issue on-site by informing participants beforehand that the best results were obtained with short gestures of the wiimote within range of the sensor bar. Secondly, many participants noted some difficulty while generating drawings because a cursor was not present on-screen to inform them about where marks would be placed. Thirdly, some participants were thrown off by their image in the final product because it did not mirror their movement in the space. Effectively, most participants desired to see themselves (and their movements) in the final product as if they were looking at themselves in a mirror. The video of the participant in the space needed to be flipped horizontally in the final product in order to produce this effect. The final issue some people noticed regarded the efficiency of the program. In response to the processing resources needed to handle the tasks of video processing and line drawing in one drawing context, the patch lagged considerably in performance. This caused the "real-time interactivity" aspect of the installation to suffer.

The evaluation of the installation at Vizagogo provided valuable information that helped this artist create a more robust governing system. As a result of the findings, some new features were added to the final iteration of the installation program. An on-screen cursor was added to help with line drawing and inherently assisted in alleviating the "large gestures" issue. Users in the space can now see a pointing location cursor on the output screen that intrinsically provides better information about the positioning limits of the wiimote. In addition, the video of the participant in the space was flipped before it was integrated with the line drawing to create a mirror effect in the final product. This artist performed a considerable amount of research to solve the lagging issue and found that making the resolution of the incoming video (from the web camera and the IR camera) half size—320x240 instead of 640x480—the frame rate of the final video doubled to over 20 frames per second. This change increased the patch's performance dramatically without losing considerable quality.

The process of this research served the purpose of establishing a workflow for artists that are working in a similar process to investigate and create new forms of interactive intermedia. Specifically, this workflow can be of assistance to artists that attempt to synchronize and harmonize visuals with real world events. Many people at Vizagogo, for example, were intrigued by the interactive elements of the installation. This artist was particularly interested in watching people's reactions when they realized that they could immerse themselves in the drawing they created based on their movements. When participants realized that the installation was more than just an application for drawing with the wiimote, it became a forum for creative discussion and techniques to push the interactive boundaries of digital media. Additionally, students of various disciplines talked with this artist about methods that were used to create this piece and how the programming environment could be used to produce work of their own on similar levels. A setting for these types of discussions is important because it facilitates an exchange of ideas and encourages new ways of thinking. This research is successful if other artists, especially those that search for ways to bridge the physical and virtual worlds, can create real-time interactive innovations based on this thesis work.

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

Producing an interactive, immersive, and customizable mark-making based real-time video installation as public art involved hours of careful planning, working on the installation's many aspects in manageable sections, improving the piece iteratively through recursive testing, and evaluating results. The workflow and techniques documented in this thesis have produced favorable results. All aspects of the installation space, such as the computing environment, sensor inputs, environment layout, lighting, and video inputs, must be thoroughly considered before constructing the major elements of the installation. Channeling the data from the sensor and video from the cameras into a program that can interpret and process this information in real-time is a crucial step that can affect the realism of interactions. The efficient integration of these sets of data and recursive testing of the product are keys to achieving an aesthetically interesting product.

6.2 Implications for Future Work

This thesis research piqued this artist's interest in and appreciation for all forms of interactive work, and may spark a similar interest in others. This process outlines only a few methods to create this type of interactive work. Artists who wish to develop interactive intermedia creations can expand upon this work by exploring alternatives for visual displays, user interactions, added user control of processes, and presentation in different venues.

The types of visual displays used for the installation can affect how the participant relates to the final visual product. A plasma screen hidden behind an ordinary picture frame, for example, could be set up in the space to display the final product and encourage the notion that everyone has an inner artist. Alternately, the visual product could be cast to an alternate surface, such as a canvas, to connect the virtual mark-making aspect to its real-world counterpart.

There are many ways to enhance interactions between the participants and the resulting imagery created in the installation space. New types of interactions can also be integrated into the existing structure to create new forms of visual output. Other unique combinations of filters could be used for processing and affected by the user's interactions within the space to obtain a unique aesthetic. Different line features or textures can be used to vary the visual quality of the drawing component and encourage more mark-making. The morphing feature of the jsui_splinestuff patch (section 4.2.2.2) could be implemented such that artists could immerse themselves in a dynamic environment with animated spline curves. Functionality for audio events such as sounds for drawing, erasing, or the z-depth position of the participant could be integrated into the work to reinforce activities occurring on-screen. Participants could also affect aspects of the visual product, such as filter parameters, by creating sounds of their own, such as claps or whistles, within the installation space.

The capabilities of the wiimote can also be exploited to add more user control in the installation. For example, the orientation of the wiimote, obtained through accelerometer readings, can be used to influence line thickness. Unused buttons on the wiimote can be used for actions such as choosing a color or paintbrush texture. The internal vibration capability of the wiimote could be used to give tactile feedback as the participant in the installation space creates a line. Incorporating a wiimote extension such as the *nunchuck*—an additional set of controls consisting of a joystick, supplementary accelerometer, and buttons that connect to the wiimote (Figure 52)—could also be utilized to add more control. A wide array of extensions exists for the wiimote, any of which can be utilized for all types of creative projects. Artists would need to take special consideration when expanding the set of controls, as too many features could bog down a participant in the installation space.

Figure 52. The nunchuck, a controller extension for the wiimote.

The thesis work, although fitting for a setting such as an art gallery, could also be implemented in other settings such as live performances, mall displays, or at entertainment venues. Music, theater, and dance performances could benefit from dimensional qualities of the visual product, which could be integrated on a stage with live elements (as in Troika Ranch's work) through one or more projections. Special consideration would need to be given to unattended public spaces in locations such as the mall or entertainment venues, as the wiimote would need to be secured or may need to be replaced with another type of interaction. Displays in these venues could encourage creativity in and add personality to an unlikely setting, reaching audiences that otherwise might not be exposed to creative vehicles.

Future work can be explored by sharing this thesis effort with the artistic community through this publication and making this work available on the internet. Specifically, this program can be made available to other Max/MSP/Jitter developers for continued improvement. In this way, I hope to inspire future aesthetic endeavors. Digital interactive media is a relatively new field, and there are many more opportunities, intermedia and otherwise, to explore the connections between the physical and virtual worlds. As technological limitations decrease, it will likely be possible in the near future to create even more impressive interactive works. With technology as a more integrated part of our everyday lives, this area of research will continue to gain increased prominence.

REFERENCES

[1] Camille Utterback. Camille Utterback :: Interactive Video Art Home Page, http://www.camilleutterback.com/, 2008.

[2] D. Takahashi. "Microsoft Unveils Surface Computer," *Design News*, vol. 62, no.10, pp. 43-46, 2007.

[3] J. Working, "The Magic Touch," *PC Magazine*, vol. 26, no. 15, p. 17, 2007.

[4] Bungie Studios, Halo 3 Home Page, http://www.halo3.com/, 2008.

[5] P. Lauber, *Painters of the Caves*, Washington D.C.: National Geographic Society, pp. 36-37, 1998.

[6] M. Capek, *Murals: Cave, Cathedral, to Street*, Minneapolis: Lerner Publishing Group, pp. 7-11, 55-61, 1996.

[7] A. Barnett, *Community Murals: The People's Art*, Philadelphia: The Art Alliance Press, pp. 11-20, 1984.

[8] E. Sullivan, *Aspects of Contemporary Mexican Painting*, New York: Americas Society, pp. 14-31, 1990.

[9] S. Goldman, *Contemporary Mexican Painting in a Time of Change*, Austin: University of Texas Press, pp. 3-14. Revised edition, 1981.

[10] S. Phillips, *Wallbangin': Graffiti and Gangs in L.A.*, Chicago: The University of Chicago Press, pp. 12-61, 309-337, 1999.

[11] Troika Ranch, Troika Ranch Home Page, http://www.troikaranch.org/, 2008.

[12] M. Coniglio and D. Stoppiello, "*About Authoring Tools: Troika Ranch,*" "Who's in Control?" New Interfaces for Artistic Expression Convention, Harvestworks Digital Media Arts Center, Eyebeam, New York, 26 February 2006.

[13] Envyloop, Envyloop Home Page, http://www.envyloop.com/, 2008.

[14] A. Arenas and J. Morris, "An Improvisory Intermedia Performance Using Live Audiovisual Sampling to Explore Mediatization as a Device of Imitative Counterpoint," *Proc. The 11th Biennial Symposium on Arts and Technology*, pp. 1-9, 2008.

[15] C. Utterback, “Unusual Positions — Embodied Interaction with Symbolic Spaces,” *First Person: New Media as Story, Performance, and Game*, N. Wardrip-Fruin and P. Harrigan, ed., Cambridge: MIT Press, pp. 218-226, 2004.

[16] E. Leffingwell, “Four Americans at Maryborough Chelsea,” *Art in America*, vol. 92, no. 9, pp. 156-157, 2004.

[17] Daniel Rozin, Daniel Rozin Interactive Art Home Page, http://www.smoothware.com/danny/, 2008.

[18] J. Bolter and D. Gromala, “Transparency and Reflectivity: Digital Art and the Aesthetics of Interface Design,” *Aesthetic Computing*. Paul Fishwick, ed., Cambridge: The MIT Press, pp. 369-382, 2006.

[19] S. Feiner, “Augmented Reality: A New Way of Seeing: Computer Scientists are Developing Systems that Can Enhance and Enrich a User’s View of the World,” *Scientific American*, vol. 286, no. 4, pp. 48, 2002.

[20] Jung von Matt, Jung von Matt/next – nextwall, http://www.nextwall.net/content/projekt/taggedinmotion.php/, 2008.

[21] Z. Lieberman, thesystemis, http://www.thesystemis.com/, 2008.

[22] Gamespot, Tiger Woods PGA Tour 2005 Q&A – GameFace II, http://www.gamespot.com/pc/sports/tigerwoodspgatour2005/news.html?sid=6104513/, 2008.

[23] Electronic Arts, Tiger Woods PGA Tour ’08 Home Page, http://www.easports.com/tigerwoods08/, 2007.

[24] C. Leung, “The Medium Is A Message,” *Canadian Business*, vol. 80, no. 13, p. 41, 2007.

[25] GestureTek, GestureTek Home Page, http://www.gesturetek.com/, 2008.

[26] G. Dayal, “Graffiti by the (Extensively Analyzed) Numbers.” *New York Times*, sec. 2: pp. 29, 25 June 2006.

[27] Graffiti Research Lab, Graffiti Research Lab >> L.A.S.E.R. Tag, http://graffitiresearchlab.com/?page_id=76#video/, 2008.

[28] TED, TED Talk: Creating Tech Marvels Out of A $40 Wii Remote, http://www.ted.org/index.php/talks/ johnny_lee_demos_wii_remote_hacks.html/, 2008.

[29] Cycling '74, Cycling '74 Home Page, http://www.cycling74.com/, 2008.

[30] T. Igoe and D. O'Sullivan, *Physical Computing: Sensing and Controlling the Physical World with Computers*, Course Technology PTR, pp. 217-248, 2004.

[31] New York Times, Wii's Means of Motion, http://www.nytimes.com/packages/khtml/2006/12/21/business/ 20061221_HOWW_GRAPHIC.html/, 2008.

[32] OSCulator, OSCulator Home Page, http://www.osculator.net/, 2008.

[33] M. Akamatsu, aka.objects, http://www.iamas.ac.jp/~aka/max/#aka_wiiremote/, 2008.

[34] D. Klipstein, Don's light, lamp, and strobe site!, http://members.misty.com/don/index.html/, 2008.

[35] *Waking Life*, Richard Linklater, dir., Louis Black, Julie Delpy, Adam Goldberg, Charles Gunning, Ethan Hawke, perf., 20th Century Fox, 2002.

[36] U. Küster, "Action Painting—Myth and Reality," *Action Painting*, F. Beyeler, ed., Germany: Ostfildern: Hatje Cantz, pp. 12-19, 2008.

[37] Cycling '74, *Jitter Tutorial*, ver. 1.6.3, pp. 438-451, 2007.

[38] Cycling '74 Forum, jsui vs. Window Listener (js + jit.window) http://www.cycling74.com/forums/index.php? t=msg&goto137367&rid=0&S=b30623700b346b17da619684d01310d0/, 2008.

[39] University of California at Berkeley, Downloads | CNMAT, http://cnmat.berkeley.edu/downloads/, 2008.

[40] J. Pelletier, cv.jit – main, http://www.iamas.ac.jp/~jovan02/cv/, 2008.

[41] K. Ohta. "Image Processing Apparatus and Storage Medium Storing Image Processing Program," U.S. Patent 2007/0211027 A1, 2007.

APPENDIX A. CALCULATING POSITION USING THE SENSOR BAR

This section illustrates how x-y position is calculated using the wiimote and sensor bar according to the wiimote's patent document.

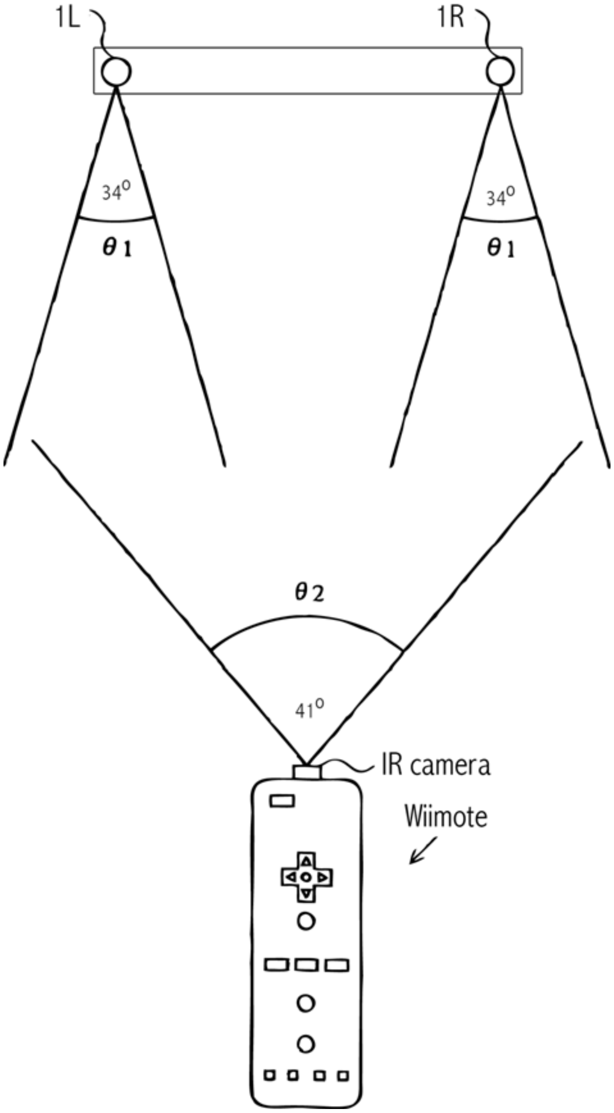


Figure 53. Position calculation using the wiimote [41].

In Figure 53, 1L and 1R are markers representing the bright IR light sources on either end of the sensor bar. Each of these markers has a viewing angle of θ_1, or approximately 34^o. When the light of both markers is within the viewing angle θ_2 (approximately 41^o) of the wiimote's IR camera, and likewise, when the IR camera is within the viewing angle θ_1 of the markers 1L and 1R, the x-y screen position of the controller can be calculated. This is computed using positional information of the markers within the wiimote's IR camera view (see Figure 54). Additionally, the distance (called "realD") of the wiimote from the sensor bar can be approximated (see Appendix B).

Figure 54. The wiimote's IR camera view of the markers used to calculate x-y screen position.

APPENDIX B. TRIANGULATION

This section illustrates how triangulation is used to calculate the wiimote's distance from the sensor bar according to the wiimote's patent document.

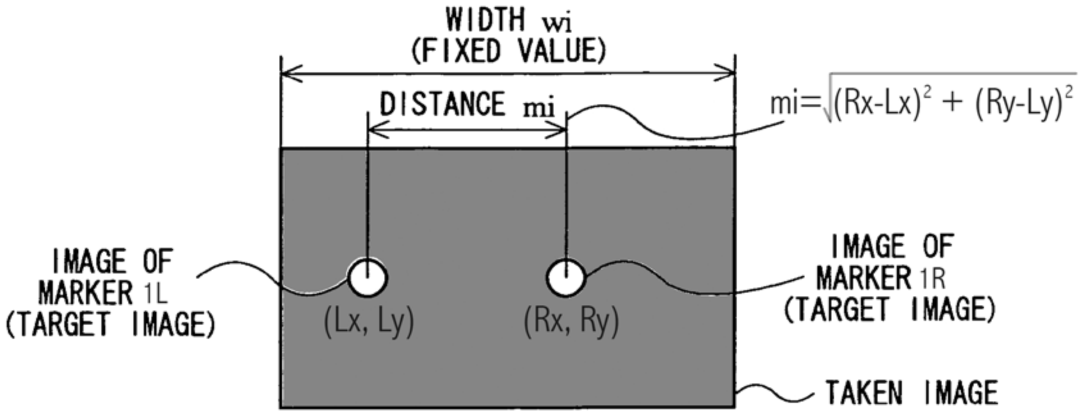


Figure 55. Distance calculation of marker points in a taken IR image [41].

Note the correlation between the markers in Figure 54 and the target images in Figure 55. First, the wiimote obtains the coordinate data of the two markers (1L and 1R) from a captured ("taken") image obtained using the wiimote's IR camera (refer to Appendix A for more information). Then, a distance (*mi*) is calculated between the two points using the standard distance formula. The fixed value *wi* represents the width of the image taken by the IR camera—a value that is pre-stored in the wiimote's CPU.

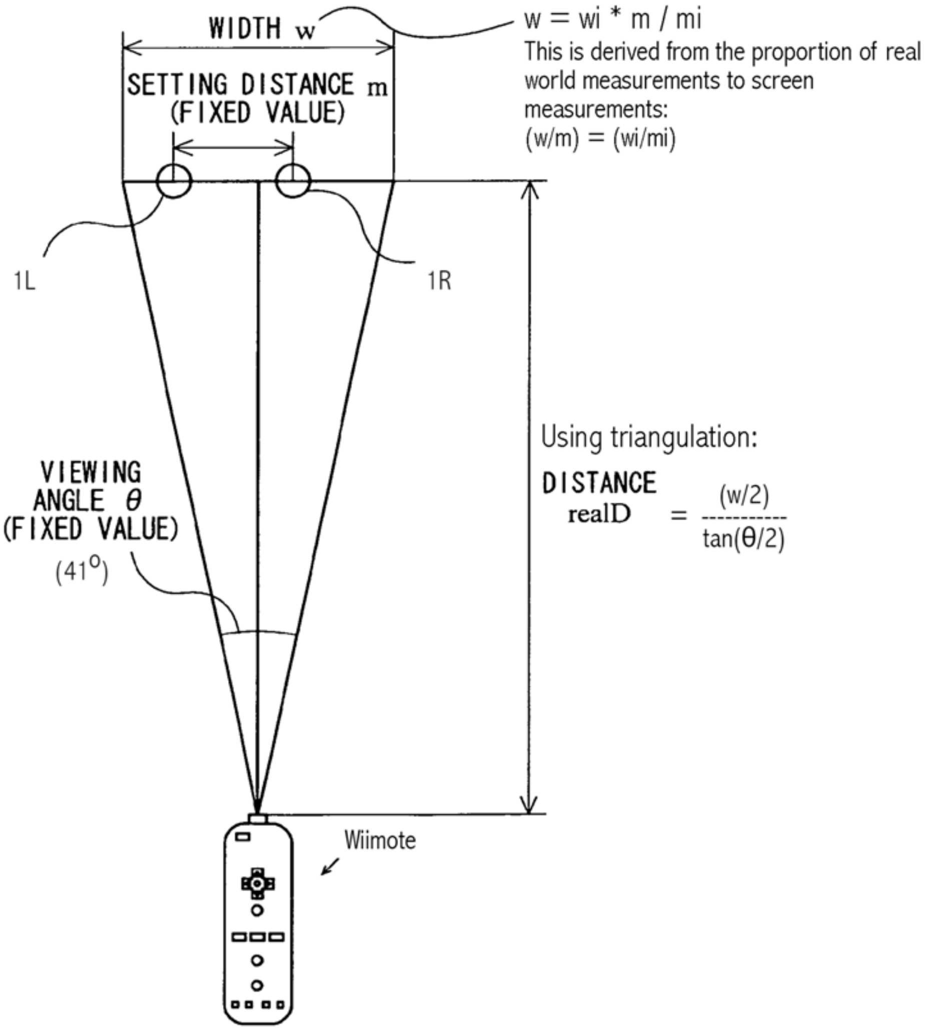


Figure 56. Calculation of realD using triangulation [41].

With these calculated values, the wiimote can compute its distance from the sensor bar (Figure 56). The value m represents the distance between the markers 1L and 1R—another fixed value that is pre-stored in the wiimote. The wiimote calculates a width w (using the formula outlined in Figure 56) that represents the (real-world) width that the IR camera is able to take in an image. The distance between the wiimote and the sensor bar ("realD", see Figure 56) is calculated using triangulation. The value w and the viewing angle θ (approximately 41^{o}) of the wiimote are used in this calculation.

APPENDIX C. JAVASCRIPT CODE

This section contains the javascript code from this artist's installation piece.

```
// LineDrawer.js

var CP_COUNT = 200;
var vx = 0;
var vy = 0;
var vz = 0;

var curpointcount = 0;
var pollpoints = 0;

//control point array
var controlX = new Array();
var controlY = new Array();
var controlZ = new Array();

//line attributes
var vslices = 20;
var vorder = 3;
var vstroketype = "basic3d";
var vrotation = [0,0,0];
var vlinesmooth = 1;
var vwireframe = 0;
var vshowpoints = 0;
var vslices = 20;
var vorder = 3;
var voutline = 0;
var voutcolor = [0,0,0,1];
var vvaryscale = 1;
var vvarycolor = 1;
var vscalerange = 0.2;
var vrotation = [0,0,0];
var vclear = 1;
var vtexture = 0;
var img = 0;

var cx = new Array();
var cy = new Array();
var cz = new Array();
var cred = new Array();
var cgreen = new Array();
var cblue = new Array();
var cscale = new Array();
```

```
//where the information from outside this script will come in
inlets = 1;

// create a [jit.window] object for our display
// (this is the object we'll "listen" to):
var mywindow = new JitterObject("jit.window","PaintStroke");
mywindow.depthbuffer = 1;
mywindow.size = [640, 480];

// create a [jit.gl.render] object for drawing into our window:
var myrender = new JitterObject("jit.gl.render", "PaintStroke");
myrender.ortho = 1;
myrender.depth_clear = 1;
myrender.depth_enable = 1;
myrender.blend_enable = 1;
myrender.doublebuffer =1;
// set background to black with full erase opacity (no trails):
myrender.erase_color = [0,0,0,1];
myrender.far_clip = 5000;

//create a [jit.gl.sketch] object for the creation of the line:
var mysketch = new JitterObject("jit.gl.sketch", "PaintStroke");
mysketch.depth_enable = 1;
mysketch.blend_enable = 1;
mysketch.glclearcolor = [0,0,0,1];
mysketch.lighting_enable = 1;
mysketch.fog = 1;
mysketch.fog_params = [0.2, 0.2, 0.2, 1., 1., 5., 15.];

//sphere controlled by the wiimote
// create a [jit.gl.gridshape] object for use to
//control with the wiimote
var mywidget = new
JitterObject("jit.gl.gridshape","PaintStroke");
mywidget.shape = "sphere";
mywidget.lighting_enable = 1;
mywidget.smooth_shading = 1;
mywidget.scale = [0.1,0.1,0.1];
mywidget.color = [1,1,1,0.5] ;
mywidget.blend_enable = 1;
mywidget.position = [0,0,0]; // no z necessary in orthographic
projection
mywidget.automatic =1;


// create an array of points generated
//based on polled mouse movement
var mycontrolpoints = new Array(CP_COUNT);
```

```
newcurve();
draw();

function triggerDown(v){
      pollpoints = v;
}

function zval(z){
      vz = z;
}

function clear(){
  //erase array
  controlX.splice(1,curpointcount);
  controlY.splice(1,curpointcount);
  curpointcount = 0;
}

function newcurve()
{
      var i;
      with (Math) {
            for (i=0;i<CP_COUNT;i++) {
                  cx[i] = random()*2-1.;
                  cy[i] = random()*2-1.;
                  cz[i] = random()*2-1.;
                  cred[i] = random();
                  cgreen[i] = random();
                  cblue[i] = random();
                  cscale[i] = random();
            }
      }
}

function draw()
{
      var type3d;

      with (mysketch) {
            reset();

            if (vclear)
                  glclear();

            // set rotation
            glmatrixmode("modelview");
            glpushmatrix();
            glrotate(vrotation[0],1,0,0); // xrotation
            glrotate(vrotation[1],0,1,0); // yrotation
```

```
glrotate(vrotation[2],0,0,1); // zrotation

gldisable("texture");

// "line_smooth" will antialias outlines
//even if fsaa is turned off
// also improves outlines when fsaa is on.
// line_smooth typically requires blending
//enabled and depth test disabled
if (vlinesmooth)
      glenable("line_smooth");
else
      gldisable("line_smooth");

if (vwireframe) // show wireframe
      glpolygonmode("front_and_back","line");
else // filled polygons
      glpolygonmode("front_and_back","fill");

type3d =  (vstroketype=="basic3d");
beginstroke(vstroketype);
//set the number of slices for each curve section
strokeparam("slices",vslices);
//set the interpolation order
strokeparam("order",vorder);
strokeparam("color",cred[0],cgreen[0],cblue[0]);

strokeparam("outline",0);

if (type3d) {
      if (voutline)
            //set the outline color
            strokeparam("outcolor",voutcolor);
      strokeparam("scale",cscale[0]*vscalerange);
}


for (i=0;i<curpointcount; i++){
      if (vvaryscale)
            strokeparam("scale",cscale[i]*vscalerange);
      if (vvarycolor)
      strokeparam("color",cred[i],cgreen[i],cblue[i]);
      if (vtexture & (curpointcount > 1))
            // 0-1 along x axis
            strokeparam("texture",i/(curpointcount-1));
      // set control point
      strokepoint(controlX[i], controlY[i],
      controlZ[i]);
}
```

```
            endstroke();

            // draw control points
            if (vshowpoints) {
                 gldisable("texture");
                 beginstroke("line");
                 strokeparam("order",1);
                 strokeparam("segments",1);
                 strokeparam("color",0.4,0.2,0,0.6);
                 for (i=0;i<vpointcount;i++) {
                      if (type2d)
                           strokepoint(controlX[i],
                           controlY[i]);
                      else
                           strokepoint(controlX[i], controlY[i],
                           0);
                 }
                 endstroke();
                 glcolor(0.2,0.6,0.2,0.6);
                 for (i=0;i<curpointcount;i++){
                      if (type2d)
                           moveto(controlX[i], controlY[i]);
                      else
                           moveto(controlX[i], controlY[i], 0);
                      circle(0.03);
                 }
            }
            // pop rotation
            glmatrixmode("modelview");
            glpopmatrix();

      }

}


function bang()
// main drawing loop...drive the renderer
{
//make sure the wiimote is in the window before drawing lines
      if(pollpoints == 1){

            var i;

            //while cur point num doesn't equal max amt for
            //curve, add to array
            if(curpointcount < CP_COUNT-1){ //!=
              i = curpointcount;
```

```
            //add poll points to control points arrays
            controlX[i] = vx;
            controlY[i] = vy;
            controlZ[i] = vz;

            //increment cur point count
            curpointcount++;

          }else{  //array has reached the end...

          }

        // move our control object to the drawing context's
        // equivalent of where our wiimote event occurred:
        mywidget.position = [vx,vy,vz];
       post("\nMouse world position is : ",
       mywidget.position[0], " ", mywidget.position[1], " ",
       mywidget.position[2]);


      }

      // rendering block...
      myrender.erase(); // erase the drawing context

      //The drawclients() method to jit.gl.render
      //collects all the relevant information from the
      //OpenGL objects attached to our drawing context and draws
      //them
      //any OpenGL objects with an automatic attribute set to 0
      //will have to be drawn manually here.
      myrender.drawclients(); // draw the client objects
     draw();

      myrender.swap(); // swap in the new drawing
}


function stroketype(v)
{
      switch (v) {
      case 1:
           vstroketype = "line";
           break;
      default:
           vstroketype = "basic3d";
      }
      draw();
}
```

```
function fullscreen(v)
// function to send the [jit.window] into fullscreen mode
{
	mywindow.fullscreen = v;
}

function ir_position(x,y){
	vx = x; vy = y;
}

function fsaa(v)
{
	mysketch.fsaa = v;
	draw();
}
```

VITA

Anna Graciela Arenas

Texas A&M University
C418 Langford Center, 3137 TAMU
College Station, TX 77843-3137

gracie.arenas@gmail.com

EDUCATION
Master of Science in Visualization Sciences Dec. 2008
Texas A&M University, College Station, TX

Bachelor of Science in Computer Science Aug. 2004
Minor in Mathematics (Cum Laude)
Texas A&M University, College Station, TX

EXPERIENCE
Electronic Arts Tiburon Summer 2007, July 2008-Present
Maitland, FL
Technical Artist (TA) – Develop tools and perform research for Tiger Woods PGA Tour game

PERFORMANCES
"Time is the Substance of Which I Am Made", Visual Artist with Jeff Morris, Eric km Clark, and Andy McWain

"Music With a View" Series Apr. 2008
Flea Theater, New York, NY

International Society for Improvised Music Dec. 2008
2nd Annual Conference
Northwestern University, Evanston, IL

PAPERS & PRESENTATIONS
11th Biennial Symposium on Arts and Technology March 2008
Ammerman Center, Connecticut College, New London, CT
"An Improvisory Intermedia Performance Using Live Audiovisual Sampling to Explore Mediatization as a Device of Imitative Counterpoint"

ASSOCIATIONS
ACM SIGGRAPH